Gradient Domain Methods for Image-based Reconstruction and Rendering

Vom Fachbereich Informatik
der Technischen Universität Darmstadt
genehmigte

DISSERTATION

zur Erlangung des akademischen Grades eines
Doktor-Ingenieurs (Dr.-Ing.)

vorgelegt von

Fabian Langguth

geboren in Sonneberg, Deutschland

Erstgutachter:	Prof. Dr.-Ing. Michael Goesele Technische Universität Darmstadt
Zweitgutachter:	George Drettakis Inria, Sophia-Antipolis

Darmstadt 2018

Langguth, Fabian: Gradient Domain Methods for Image-based Reconstruction and Rendering
Darmstadt, Technische Universität Darmstadt
Jahr der Veröffentlichung auf TUprints: 2018
Tag der mündlichen Prüfung: 02.07.2018

Erklärung zur Dissertation

Hiermit versichere ich die vorliegende Dissertation selbständig nur mit den angegebenen Quellen und Hilfsmitteln angefertigt zu haben. Alle Stellen, die aus Quellen entnommen wurden, sind als solche kenntlich gemacht. Diese Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Darmstadt, den 18. Mai 2018

Fabian Langguth

Abstract

This thesis describes new approaches in image-based 3D reconstruction and rendering. In contrast to previous work our algorithms focus on image gradients instead of pixel values which allows us to avoid many of the disadvantages traditional techniques have. A single pixel only carries very local information about the image content. A gradient on the other hand reveals information about the magnitude and the direction in which the image content changes. Our techniques use this additional information to adapt dynamically to the image content. Especially in image regions without strong gradients we can employ more suitable reconstruction models and we can render images with less artifacts. Overall we present more accurate and robust results (both 3D models and renderings) compared to previous methods.

First, we present a multi-view stereo algorithm that combines traditional stereo reconstruction and shading based reconstruction models in a single optimization scheme. By defining as gradient based trade off our model removes the need for an explicit regularization and can handle shading information without the need for an explicit albedo model. This effectively combines the strength of both reconstruction approaches and cancels out their weaknesses.

Our second method is an image-based rendering technique that directly renders gradients instead of pixels. The final image is then generated by integrating over the rendered gradients. We present a detailed description on how gradients can be moved directly in the image during rendering which allows us to create a fast approximation that improves the quality and speed of the integration step. Our method also handles occlusions and compared to traditional approaches we can achieve better results that are especially robust for scenes with reflective or textureless areas.

Finally, we also present a new model for image warping. Here we apply different types of regularization constraints based on the gradients in the image. Especially when used for direct real-time rendering this can handle larger distortions compared to traditional methods that use only a single type of regularization.

Overall the results of this thesis show how shifting the focus from image pixels to image gradients can improve various aspects of image-based reconstruction and rendering. Some

of the most challenging aspects such as textureless areas in rendering and spatially varying albedo in reconstruction are handled implicitly by our formulations which also leads to more effective algorithms.

Zusammenfassung

Diese Arbeit bescheibt neue Verfahren für bildbasierte 3D Rekonstruktion und Rendering. Anders als vorhergehende Arbeiten fokusieren wir unsere Algorithmen auf die Ableitungen der Pixel im Bild anstatt die Pixel selbst. Das erlaubt es uns viele der Schwächen von traditionelleren Verfahren zu umgehen. Ein einzelnes Pixel trägt nur sehr wenige und sehr lokale Informationen über den Bildinhalt. Die Ableitung allerdings ermöglicht direkte Rückschlüsse auf die Art und Richtung in der sich der Bildinhalt ändert. Unsere Methoden nutzen diese Information, um sich dynamisch an den Bildinhalt anzupassen. Vor allem in Bildregionen mit sehr wenigen starken Veränderungen können wir dadurch effektivere Algorithmen entwickeln und einsetzen, und die Qualität der Rekonstruktionen und der gerenderten Bilder erhöhen. Unsere Methoden sind robuster und die Ergebnisse zeigen zudem weniger Artefakte im Vergleich zu vorhergehenden Verfahren.

Als Erstes beschreiben wir einen neues Multi-view Stereo Verfahren das traditionelle geometrische Formulierungen mit auf Schattierung bassierten Modellen in einer kombinierten Optimierung verbindet. Wir wechseln unser Model dynamisch zwischen den beiden Ansätzen basierend auf den Gradienten im Bild und können so ihre Stärken kombinieren und die Schwächen entfernen. Dies führt dazu das wir keine explizite Regularisierung mehr benötigen und auch eine wechselnde Oberflächenfarbe nicht mehr explizit modelieren müssen.

Unsere zweite Methode ist ein bildbasierter Rendering Algorithmus, der keine Pixel sondern direkt Gradienten nutzt, um neue Bilder zu generieren. Nachdem das Bild aus den Gradienten erstellt wurde, wird es integiert, um es wieder darstellen zu können. Wir beschreiben außerdem wie man Ableitungen direkt auf dem Bild verschieben kann, um so eine schnelle Annäherung zu generieren, die das Integrieren beschleunigt. Unsere Methode ist auch in der Lage, mit Verdeckungen umzugehen, und sie kann besonders reflektierende Oberflächen besser erfassen als andere Verfahren.

Unsere letzte Methode basiert auf Bildwarping. Hier wird der Bildinhalt direkt modifiziert, allerdings verwenden wir wiederum unterschiedliche Regularisierungsbedingungen basierend auf der Stärke der Gradienten im Bild. Verglichen mit vorhergehenden Methoden, die nur eine bestimmte Art der Regularisierung verwenden, können wir dadurch größere Veränderungen mit weniger Artefakten darstellen.

Zusammenfassend zeigt unsere Arbeit dass die Verwendung von Bildgradienten für verschiedene Methoden aus dem Bereich der bildbasierten Rekonstruktion und des Renderings deutliche Vorteile bringt. Einige der schwierigsten Aspekte werden von unseren Modellen implizit behandelt, was zu besseren und effektiveren Algorithmen führt.

Acknowledgements

I would like to thank my supervisor Prof. Dr.-Ing. Michael Goesele for his support and feedback as well as George Drettakis who kindly agreed to review this thesis. I would also like to thank my mentors during my internships at Adobe and Facebook: Kalyan Sunkavalli, Sunil Hadap, and Johannes Kopf.

Special thanks go out to my colleagues and friends that I worked with over the years: Jens Ackermann, Simon Fuhrmann, Ronny Klowsky, Sven Widmer, Dominik Wodniok, Martin Hess, Daniel Thuerck, André Schulz, Nicolas Weber, Stefan Guthe, Jürgen Bernard, Mate Beljan, Patrick Seemann, Nils Möhrle, Stepan Konrad, Daniel Thul, Carsten Haubold, Sebastian Koch, Patrick Mücke, Aleksander Hołyński, Clément Godard, and Jacques Cali.

Finally I want to thank my family for their support during all those years.

This work was also supported in part by the European Commission's Seventh Framework Programme under grant agreements no. ICT-611089 (CR-PLAY).

Contents

Chapter 1

Introduction

Over the past years virtual and augmented reality have become major topics for both scientific research and consumer products. Many consider these technologies to become one of the most important ways to interact with computers in the future. A variety of applications like virtual workplaces, tourism, games, and social media promise immersive experiences (see Figure 1.1). For many of these applications it is crucial to create and display realistic

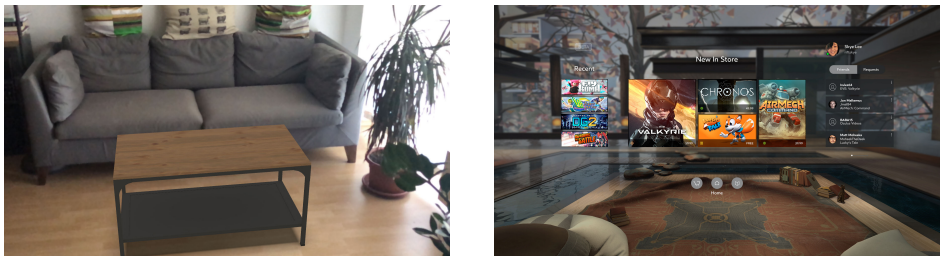


Figure 1.1: Example experiences for AR and VR. *Left*: A virtual object is inserted into a real scene using an AR toolkit on a mobile phone. *Right*: A completely virtual home is rendered in VR with games and social information added to the experience (image © Oculus VR https://en.oculusbrand.com/).

digital content. For this purpose traditional computer graphics rendering requires detailed models of the scene geometry, texture, and lighting. In many applications these models are manually generated by artists which is a very long and meticulous process. A more practical approach would be to capture all of these aspects from real world objects and scenes and use this data to create a synthetic representation. This is however also a very hard and complex problem. Acquiring high-quality geometry requires sophisticated reconstruction processes and while active techniques such as structured light [Salvi et al., 2010; Ritz et al., 2012] and laser scanning [Vosselman and Maas, 2010] can achieve excellent results, they require a complex setup or are too expensive to be used in a broader consumer setting. Many advancements therefore push towards image-based technologies that are built on computer

Figure 1.2: Problems with previous methods. *Top*: An image and a 3D reconstruction by Zollhöfer et al. [2015]. The change in surface color leads to artifacts in the reconstruction (see the dark dot on the object). *Bottom*: An image and a rendering by Sinha et al. [2012]. Here the explicit separation of the reflection is not accurate enough and leads to render artifacts.

vision algorithms. Camera sensors have become very energy efficient, small, and ubiquitous. Almost every smartphone already includes a high-quality camera which can make the acquisition of image-based content straightforward for many people. But while images and videos can be captured very easily with modern devices, using them for geometry reconstruction and rendering remains a challenging problem and a very active research topic in both computer vision and graphics.

Over the past years many advancements have been made towards dealing with more general imagery or increasing the overall quality of the output. Approaches like Snavely et al. [2006] and Goesele et al. [2007] leverage large internet photo collections for image-based modeling and reconstruction. Other techniques like Wu et al. [2011c] use additional reasoning about the natural scene lighting and shading to recover very fine geometry details from images. Chaurasia et al. [2013] increase the robustness of image-based rendering for scenes that have complex properties and are hard to reconstruct. While these and other methods have pushed the boundaries and made image-based techniques available for more uncontrolled settings they still have limitations. The reconstruction of scenes with low amounts of texture often leads to artifacts or low quality results while reflective objects also pose challenges

for rendering algorithms. On the other hand, a varying surface color can affect refinement algorithms like Wu et al. [2011c] and prevent them from operating properly. Some algorithms try to explicitly deal with these problems. Zollhöfer et al. [2015] model surface color in their geometry refinement while Sinha et al. [2012] separate reflective layers to handle more complex scenes. While these approaches can improve upon previous techniques their models are based on certain heuristics that fail in some scenarios, see Figure 1.2.

In this thesis we introduce novel techniques that further advance the state of the art in both image-based 3D reconstruction and rendering. The key idea of our work is to focus on image gradients instead of pure pixel values. Gradients have the inherent property to reveal the important changes in scene geometry or lighting and we aim to use this observation to create more detailed 3D geometry and more robust renderings. Most importantly we will show that using gradients leads to formulations that treat difficult aspects of reconstruction and rendering implicitly. We therefore no longer need to create explicit models of surface color or reflective layers like Zollhöfer et al. [2015] and Sinha et al. [2012]; our approaches simply handle these properties automatically. Specifically, we develop a new multi-view stereo algorithm that can create detailed geometry reconstructions by using shading information. Here our gradient-based formulation allows us to be robust against changes in the surface color. We also show how image-based rendering can be done using solely image gradients. This approach automatically handles reflections without the need to separate them from a reflective surface. Finally we also present an image warping technique that can be used for fast image-based renderings without additional precomputation. Here we use image gradients to create a more robust regularization that improves the ability of image warps to extrapolate the input data.

1.1 Multi-view Stereo

Image-based reconstruction methods usually employ multi-view stereo to generate high quality results [Goesele et al., 2007; Hiep et al., 2009; Furukawa and Ponce, 2010]. However, these methods typically operate on larger regions of image pixels and/or use surface regularization in order to be robust to noise. As a result, they often cannot recover fine-scale surface details accurately. Another branch of recent work has therefore focused on shading-based refinement of the geometry obtained from multi-view stereo (or in some cases using depth sensors or template models). The shading of an object can usually reveal much more detail but is also harder to model as it is affected by both lighting in the scene and the objects reflectance properties. Early work in this domain could only be used for objects with constant albedo [Wu et al., 2011c], but recently algorithms have also evolved to operate more complex objects [Zollhöfer et al., 2015]. All the methods presented so far treat the coarse input geometry as a fixed ground truth estimate of the shape and use it to regularize their

refinement optimization. Consequently, uncertainties in the initial reconstruction method are discarded and cannot be resolved reliably. Another challenge for these shading-based techniques is that observed image intensities combine shading and surface albedo. Inferring fine-scale detail from shading thus requires an explicit reasoning about surface albedo. This significantly increases the number of variables in the optimization and requires additional heuristics for regularization. Most current techniques either assume constant albedo or apply strong regularization on the albedo distribution, which can often fail on real-world surfaces. We propose a new multi-view surface reconstruction approach that combines stereo and shading-based data terms into a single optimization scheme using image gradients. At the heart of our algorithm is the observation that stereo-matching and shape-from-shading have complementary strengths. While stereo correspondences are more accurate in regions with many large image gradients, shape-from-shading is typically more robust in flat regions with no albedo variations. By using a simple image gradient-based trade-off between stereo and shading energies we can maximize their effectiveness. The resulting algorithm provides distinct advantages over previous work. It combines multi-view stereo and shading-based reconstruction, balancing the two terms without committing, a priori, to either of them. And, due to this balancing, it can treat spatially varying albedo implicitly, i.e. the optimization is robust against spatially varying albedo without explicitly modeling it. This allows us to completely remove any form of regularization from our energy formulation and introduce shading-based terms that are solely based on geometry.

1.2 Image-based Rendering

Many of the problems of multi-view stereo also pose challenges to image-based rendering. It is hard to reliably recover scene depth in poorly textured areas or scenes with complex reflectance properties. Stereo methods generally only recover a single depth value per input pixel. Reflective or glossy scenes cannot be rendered with this simple model as pixel values are usually a combination of several separate layers. Some approaches already tried address this issue for scenes with reflections by decomposing each input photo into multiple layers [Sinha et al., 2012], each with their own estimated depth. However, it is very hard to recover accurate depth information and separations for all layers and this approach often generates various artifacts. Observing the previously described strengths of stereo depending on image gradients, we carry over the idea from our multi-view stereo algorithm and introduce a new approach to image-based rendering that operates in the gradient domain. Here we interpret the result of a stereo reconstruction as the depth of the image *gradients* rather than *pixels*. We then project the gradients to their new locations as seen from the novel viewpoint and reconstruct the image through Poisson integration. To provide a data term for the Poisson solver, we directly render the effect of a gradient moving within the image. We show that this

approach has many advantages over traditional image-based rendering. Most importantly, in less textured regions, where the depth from stereo algorithms is less reliable, the gradients are very small and do not contribute much to our final results. Also, due to the fact that gradients in natural images are sparse, our approach provides a natural separation of various potential layers, as every gradient is usually generated by only a single layer. The approach is therefore particularly well suited for handling reflections and other non-Lambertian effects.

1.3 Image Warping

While our rendering method itself can operate in realtime it still requires an expensive preprocessing step to reconstruct scene geometry. We therefore also present another more lightweight approach that builds only on a sparse reconstructions and uses image warping instead of full 3D rendering. The concept of image warping is traditionally used for a variety of non-interactive tasks such as image retargeting (resizing, rotating) [Zhang et al., 2009; He et al., 2013], or video stabilization [Liu et al., 2009]. To utilize image warping for rendering we therefore specifically focus on data extrapolation, which has not been discussed in previous work. We show how warping constraints can be modified to allow for more extrapolation in image regions without salient content. Following the observations from our previous method, gradients are a great indicator. Image regions without strong gradients can absorb much more distortion and we show that our warping method can contain these distortions in much more local regions compared to previous work.

1.4 Thesis Outline

The rest of this thesis is organized as follows: We first describe the theoretical background on relevant computer vision and graphics topics in Chapter 2. Next Chapter 3 covers our contributions in the domain of multi-view stereo and describes our shading-aware approach. The contents of this chapter were previously published as [Langguth et al., 2016]. Chapter 4 discusses our gradient domain image-based rendering approach which is derived from our publication [Kopf et al., 2013]. Chapter 5 introduces our new regularization approach for image warping. The thesis ends with a conclusion and an outlook on future work in Chapter 6.

Chapter 2

Background

This chapter introduces basic concepts and algorithms that are essential to image-based 3D reconstruction and rendering. We will describe the main ideas of image formation and camera geometry, and how they can be utilized to recover the 3D structure of a real world scene from a set of captured images. The next section then covers the principles of multi-view stereo which is used to reconstruct dense geometry from a set of images with calibrated camera information. Finally we describe classic techniques for image-based rendering which often also builds on calibration and reconstruction steps. For a wider and more in-depth view of all fields please also see the relevant books and articles [Hartley and Zisserman, 2004; Kang et al., 2006; Szeliski, 2010]. In the later chapters of the thesis we will then also discuss recent advancements in the respective fields and how they relate to our methods.

2.1 Geometric Image Formation

The formation of an image in a digital camera is a complex process that involves many geometric and photometric relationships. Many of these depend on involved physical processes and cannot be simulated to arbitrary accuracy. In computer vision and graphics it is therefore common to use simpler abstractions that are computationally inexpensive but still model real world interactions accurately enough to represent the basic physical properties of light. Figure 2.1 shows an abstraction of the image formation. The value that is captured at a single image pixel depends on a light ray that is emitted from a light source and reflected into the camera by a surface in the scene. The photometric parameters such as the intensity of the light source and the normal and reflectance properties of the surface determine the amount of the light arriving at the image sensor. The position, orientation, and optical parameters of the cameras determine the image pixel location where a certain surface element is observed.

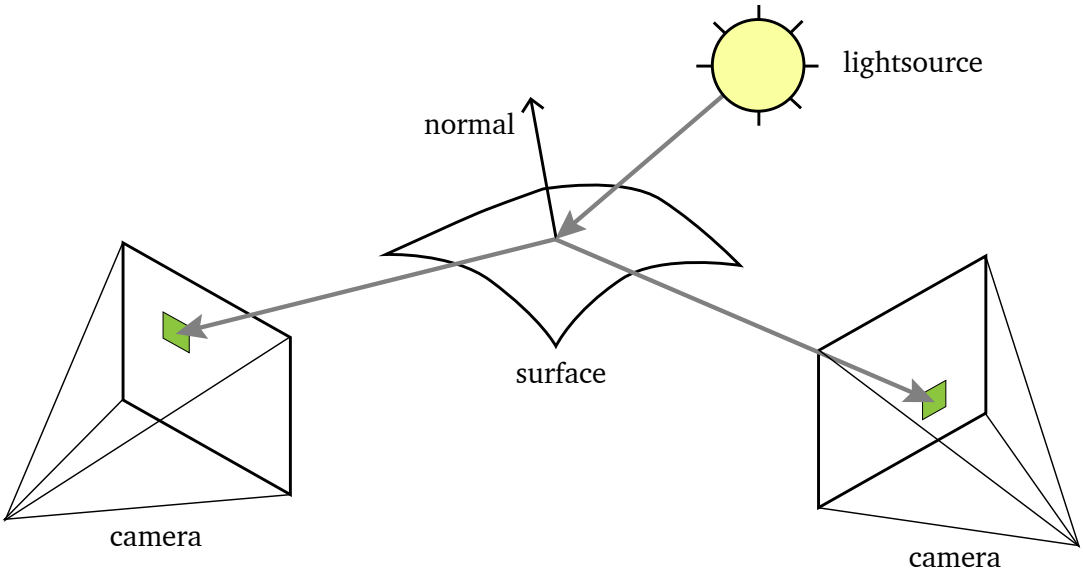


Figure 2.1: A general model of image formation in computer vision. Light, that is emitted from a source, travels as a ray through the scene, is reflected by a surface and eventually reaches the cameras image sensors at certain pixels.

2.1.1 Camera Projection

Even though modern cameras have very complicated lens systems they can be approximated to a large extent with a pinhole model [Hartley and Zisserman, 2004]. With this model the pixel location of a surface element can be determined with linear and projective transformations. Given a point in space, it's 3D coordinate $\bar{\mathbf{x}}$ is first transformed into a local camera coordinate system. This depends on the center $\mathbf{c}$ and orientation $\mathbf{R}$ (represented as a 3D rotation matrix) of the camera. It can be written as a 3D translation, moving the camera center to the origin, followed by a rotation, aligning the optical axis of the camera with the z-axis of the coordinate system:

$$\mathbf{x}' = \mathbf{R}(\bar{\mathbf{x}} - \mathbf{c}) \tag{2.1}$$

It is common to formulate this as a combined matrix multiplication with a homogeneous coordinate by introducing a specific translation component $\mathbf{t} = -\mathbf{R}^T\mathbf{c}$:

$$\mathbf{x}' = [\ \mathbf{R}\ \mathbf{t}\ ]\bar{\mathbf{x}} \tag{2.2}$$

The resulting point can now be projected onto the image plane using true 3D perspective by dividing the coordinate by it's z component. The resulting coordinate $\hat{\mathbf{x}}$ is therefore given by

$$\hat{\mathbf{x}} = \mathbf{x}'/\mathbf{x}'_z = \begin{bmatrix} \mathbf{x}'_x/\mathbf{x}'_z \\ \mathbf{x}'_y/\mathbf{x}'_z \\ 1 \end{bmatrix}. \tag{2.3}$$

The z component is also called the depth of the point and stereo methods aim to recover this value to create a full 3D reconstruction. The last remaining step is to convert this coordinate to actual pixel coordinates in an image. These coordinates are usually measured between $(0,0)$ and (W,H) where W and H are the image width and height in pixels. This transformation depends on the camera *intrinsics*. The intrinsics describe the relation of the image sensor to the projection center of the camera. They are usually written in the form of an upper-triangular matrix

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.4}$$

f_x and f_y represent the focal length of the camera, the distance of the image plane to the camera center in units of the pixel size. s is the skew which measures a misalignment when the image sensor is not mounted orthogonal to the optical axis of the camera. The point (c_x, c_y) is called the principal point and describes the pixel coordinate where the optical axis intersects the image plane. For modern cameras it is safe to assume a very high build quality which allows us to make more simplifications. Pixels are always square so there is only one focal length $f = f_x = f_y$, and the camera construction is accurate enough to assume a skew of 0, which leaves us with a simplified matrix

$$\mathbf{K} = \begin{bmatrix} f & 0 & c_x \\ 0 & f & c_y \\ 0 & 0 & 1 \end{bmatrix}. \tag{2.5}$$

It is also safe to assume that the principal point is always at the image center so

$$(c_x, c_y) = (W/2, H/2). \tag{2.6}$$

The $\mathbf{K}$ matrix does not affect the z-component of the coordinate so it can be combined with the camera extrinsics to form a general camera matrix $\mathbf{P}$ which directly maps from world coordinates to image coordinates

$$\mathbf{P} = \mathbf{K} \, [\, \mathbf{R} \; \mathbf{t} \,]. \tag{2.7}$$

The 2D image coordinate $\mathbf{x} = (x, y)$ is then given by

$$x = \frac{\mathbf{P}_0 \bar{\mathbf{x}}}{\mathbf{P}_2 \bar{\mathbf{x}}}, \quad y = \frac{\mathbf{P}_1 \bar{\mathbf{x}}}{\mathbf{P}_2 \bar{\mathbf{x}}} \tag{2.8}$$

where $\mathbf{P}_i$ is the i-th row of $\mathbf{P}$.

Figure 2.2: Example for radial lens distortion. *Left*: Original image. *Right*: Undistorted image. Even though the radial distortion is not very high and mostly effects the border of the image, it can have a large impact on the final reconstruction quality. It is therefore essential to take it into account in reconstruction algorithms.

2.1.2 Radial Distortion

As noted before the pinhole model is only an approximation of a real camera optics and many lenses, especially if they have a small focal length, introduce an additional radial distortion. This is a nonlinear effect that cannot be approximated with linear algebra. However it can still be accounted for during camera calibration, see Section 2.3. Depending on the lens various models have been developed [Kannala and Brandt, 2006] but for regular consumer cameras the most commonly used distortion model is a low-order polynomial. This model has a low number of parameters that is easy to estimate during calibration but also powerful enough to handle the most common distortions. Let $\hat{\mathbf{x}} = (x, y)$ be the image coordinate after the perspective division but before applying the $\mathbf{K}$ matrix, we can formulate the distortion using the radius $r^2 = x^2 + y^2$ as

$$\begin{aligned} x_d &= x(1 + \kappa_1 r^2 + \kappa_2 + r^4) \\ y_d &= y(1 + \kappa_1 r^2 + \kappa_2 + r^4) \end{aligned} \tag{2.9}$$

where κ_1 and κ_2 are called the radial distortion parameters. Once the parameters are estimated it is possible to create an *undistorted* version of the image by reverting the effects of the distortion, see Figure 2.2. The undistorted image directly conforms to the camera matrix $\mathbf{P}$ and can be used for further processing.

2.2 Photometric Image Formation

Having determined where a 3D point is observed in an image we also need to model the intensity or color we can expect at the pixel location of the point. This value depends on the material properties or reflectance of the surface and the lighting in the scene. Both

reflectance and lighting can be very complex and depend on many factors. For the purpose of this thesis we will focus on a restricted model and refer to more extensive articles for further details [Glassner, 1994; Weyrich et al., 2009].

2.2.1 Lighting and Reflectance

All light in a scene originates from a light source and is then reflected by various surfaces. It is possible to model these light sources explicitly as point or area light sources [Cohen et al., 1993], but capturing realistic models from real world data requires complex setups [Goesele et al., 2003]. For our purposes it is sufficient to only model the incident illumination at a certain point in space as this is sufficient to represent the lighting on a single object in a scene. This illumination is usually represented as an *environment map* [Blinn and Newell, 1976; Greene, 1986], $L_i(\mathbf{v}_i;\lambda)$, which maps incident light directions $\mathbf{v}_i$ to intensity values for a certain color (or wavelength) λ. It is possible to capture an environment map directly by taking an image of a mirrored sphere [Debevec, 2008], but, as described shortly, this model can be approximated even further and we will later recover coarse lighting information from an image without a priori knowledge or complex equipment. A general model of light transport has to simulate complex physical interactions and, as noted before, we focus on a greatly simplified version. First we only model objects that do not emit light themselves. Second we assume that all objects are completely opaque. This means that all incident light at a given point on the surface is either reflected or absorbed, but no light transport happens inside the object. Second we assume that the surface is *isotropic*, i.e. there is no preferred reflectance orientation along the surface (brushed metal is an example of a material that is *not* isotropic). In this scenario we can use a *bidirectional reflectance distribution function* (BRDF) $f_r(\mathbf{v}_i,\mathbf{v}_r,\mathbf{n};\lambda)$ [Nicodemus et al., 1977]. It describes how much light from an incident direction $\mathbf{v}_i$ is emitted in a reflected direction $\mathbf{v}_r$ depending on the surface normal $\mathbf{n}$. A pixel in a camera observes light from a single reflected direction, $L(\mathbf{v}_r;\lambda)$, for each surface point, but the incident lighting has to be integrated over all possible directions, leading to

$$L_r(\mathbf{v}_r;\lambda) = \int L_i(\mathbf{v}_i;\lambda)\, f_r(\mathbf{v}_i,\mathbf{v}_r,\mathbf{n};\lambda)\, cos\theta_i\, d\mathbf{v}_i \tag{2.10}$$

The angle θ_i is measured between the incident direction and the surface normal and the cosine factor $cos\theta_i = \mathbf{v}_i \cdot \mathbf{n}$, also called *foreshortening* factor, models the surface area that is exposed to the incoming light from a certain direction. At oblique angles this area increases leaving less light to be reflected at a single point.

For our reconstruction approach (Chapter 3) we cannot estimate an object's BRDF completely. We will therefore make another simplification and assume only *diffuse* or *Lambertian* reflectance. Here light is reflected uniformly in all directions which leads to a BRDF that only models the color $f_r(\mathbf{v}_i,\mathbf{v}_r,\mathbf{n};\lambda) = f_r(\lambda)$. This color is also called *albedo* or *diffuse albedo*. The

amount of reflected light now only depends on the incident light, the albedo, and the surface normal.

$$L_r(\lambda) = \int L_i(\mathbf{v}_i;\lambda)\, f_r(\lambda)\, cos\theta_i\, d\mathbf{v}_i \tag{2.11}$$

Spherical Harmonics A key observation in computer graphics was that for diffuse objects and distant illumination, environment maps can be represented very compactly using spherical harmonics [MacRobert and Sneddon, 1967] which are a form of frequency decomposition on a sphere. As shown by Ramamoorthi and Hanrahan [2001a] diffuse reflectance is limited to mostly low frequencies and therefore the incident light can be approximated with only the first few spherical harmonics bands. The integral over all incident light then becomes a simple sum of spherical harmonics basis functions and their coefficients.

$$L_r(\lambda) = \sum_h B_h(\mathbf{n})\mathbf{l}_h(\lambda) \tag{2.12}$$

The number of components h depends on the number of spherical harmonics bands that are used. Most common are 3 or 4 bands, leading to 9 or 16 parameters. With this approximation it is possible to recover a realistic model of the incident illumination from a single image of an object with known geometry [Ramamoorthi and Hanrahan, 2001b].

2.2.2 Digital Cameras and Images

Given a model of how much light is emitted by a certain surface the remaining question is how this light is translated into pixel values in a digital camera. As mentioned before the light passes through a lens system and is then projected onto the image plane where a sensor measures the amount of light. While there are different types of sensor (most commonly used are CCD and CMOS) they all follow a basic physical principle. When light hits a certain pixel on the sensor it induces a small electrical charge. This charge is accumulated over a certain amount of time, the *exposure time*, and has a direct linear relation to the amount of light that hits the sensor. After reading and amplifying the charges from the whole sensor the information can be translated into an image. As the overall power levels generated by the light are very small this process is susceptible to generating noise - especially for scenes without a lot of light where the signal needs to be amplified significantly.

Generally the image sensor measures light from all visible frequencies simultaneously, so the generated pixel value is actually an integral over all emitted wavelengths $\int L_r(\lambda)d\lambda$. To distinguish between different colors, filters are added on top of the sensor and limit the range of wavelengths for each pixel. The filters usually have one of 3 different colors: red, green, and blue - following the human color perception - and the most common arrangement

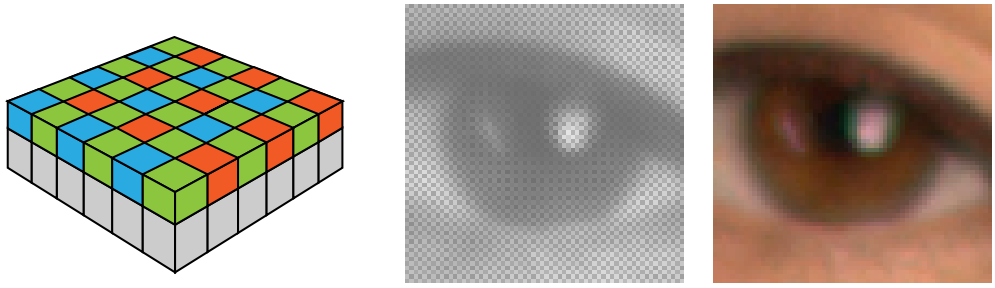


Figure 2.3: *Left*: Color filters on top of the image sensor arranged as a Bayer pattern. There are twice as many green pixels compared to red and blue as humans are more sensitive to green light. *Center*: Raw image data; every pixel measures the intensity of one color channel revealing the Bayer pattern. *Right*: For the final image the two missing color values for every pixel are interpolated to create 3 colors for every pixel.

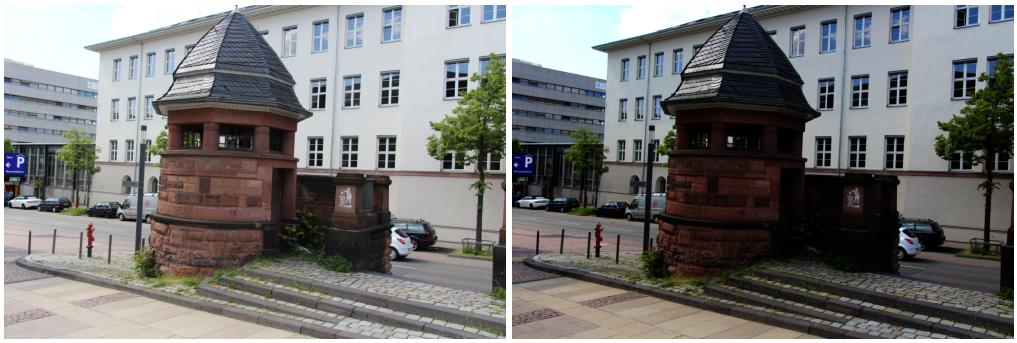


Figure 2.4: *Left*: Gamma corrected image that is generated to be displayed on a screen. *Right*: Image without gamma correction. This version has a linear response to the actual light in the scene which is required for many computer vision models.

is the Bayer pattern, see Figure 2.3. With this pattern every pixel measures a single color and the missing color values are later interpolated for the final image. The exact concepts behind human color perception and different digital color spaces are described in detail by Fairchild [2013].

The final pixel values are stored in RAW images without modification, or they are directly optimized for display. In the latter case they are usually quantized and compressed to save storage space, and it is also very common to directly apply a gamma correction to the image. This function maps the previously linear response of the sensor into a non-linear space that is similar to the human perception of brightness. While this is necessary to increase the visual quality of the image it is problematic for computer vision. For the reconstruction of scene reflectance and lighting it is important to first remove the gamma correction from the image, see Figure 2.4. An even better way is to estimate the actual response function

of the camera [Debevec and Malik, 1997; Robertson et al., 2003] which can reveal further non-linear processing.

2.3 Geometric Camera Calibration

The first step in any image-based pipeline is camera calibration. If we want to reconstruct depth information we first need to know the projection functions, i.e the **P** matrices, of all input cameras. This can be done from a number of 3D to 2D correspondences. Starting from Equation (2.8) we can formulate a linear least squares problem that can be readily solved for **P**. In controlled setups special targets with known geometry and known texture are used to directly create these correspondences. Unfortunately for images from general, unknown scenes these correspondences are not immediately available. In this case the camera parameters and the 3D structure of the scene have to be estimated simultaneously. This process is usually referred to as *structure from motion* or *SfM*. SfM first establishes 2D to 2D correspondences between pairs of images. These correspondences can then be used to estimate relative transformations between camera pairs. It is possible to merge all pairwise relations into a globally consistent reconstruction [Wilson and Snavely, 2014] but a more robust solution is to create an incremental reconstruction [Schönberger and Frahm, 2016]. Here the poses of an initial camera pair are used to estimate the 3D positions of all 2D-2D correspondences via triangulation. The subset of correspondences that maps into another image can then be used to directly estimate the projection matrix of the next camera. Subsequent iterations can then triangulate even more correspondences and incrementally add more cameras. This process could potentially also lead to an accumulation of errors due to noise in the input data. It is therefore necessary to adjust the reconstruction globally every few iterations, which is usually done with a global non-linear optimization called *bundle adjustment*. The final output of SfM is then a set of intrinsic and extrinsic parameters for every camera along with a sparse set of 3D points and their correspondences to specific views.

Open Source Framework All of the techniques described in this section have been implemented in an open source framework [1] in parallel to the research that was performed for this thesis. The details of the framework are outlined by Fuhrmann et al. [2014] and Fuhrmann et al. [2015]. All of the software is completely free under a permissive licence and the reconstruction approach we will describe in Chapter 3 also directly integrates into this framework.

[1] https://github.com/simonfuhrmann/mve

2.3.1 Feature Matching

Finding 2D to 2D correspondences is an essential problem in computer vision as it is required for a variety of tasks. The best approach for SfM is to find and match image *feature points* (also called *interest points* or *keypoints*). The main idea is that features are first detected as specific image locations that have high local contrast or gradients [Shi and Tomasi, 1994; Triggs, 2004] and are then compared against features in other images using a descriptor that is build from a patch of pixels around the feature location. As the appearance of objects can change across images due to perspective, scale, or lighting differences, both detection and descriptor matching have to be robust against these changes. The most common and successful algorithms are SIFT [Lowe, 2004] and SURF [Bay et al., 2008] as they deal with all of these challenges. Figure 2.5 shows an example of two images, their detected SIFT features, and the final matches obtained with SIFT. It is immediately clear that these matches contain false positives and one of the challenges of SfM is to deal with these outliers properly.

2.3.2 Two View Geometry

We can describe a special form of geometry for a single pair of cameras that allows us to recover their relative rotation and translation from a set of features matches. We will only describe the core principle, a detailed coverage of the topic can be found in Hartley and Zisserman [2004]. Consider a single 3D point $\bar{\mathbf{x}}$ that is projected into two cameras. We can assume, without loss of generality, that the first camera is located at the origin and aligned with the z-Axis, so its rotation matrix is the identity $\mathbf{I}$ and its translation is 0. The relative transformation of the second camera is then given by a rotation $\mathbf{R}$ and a translation $\mathbf{t}$. We know from Section 2.1 that the coordinate $\hat{\mathbf{x}}$ is a simple projection of $\bar{\mathbf{x}}$ onto the image plane and thus lies on a ray from the camera center to the 3D point. This means that the vectors from the camera centers to the image plane coordinates of the 2D features and the vector connecting both camera centers have to be inside of a single plane which is called the *epipolar plane*, see Figure 2.6. Formally we get

$$\hat{\mathbf{x}}_1 \cdot (\mathbf{t} \times \mathbf{R}\hat{\mathbf{x}}_0) = \hat{\mathbf{x}}_1^T [\mathbf{t}]_\times \mathbf{R}\hat{\mathbf{x}}_0 = 0 \tag{2.13}$$

which leads to a matrix $\mathbf{E} = [\mathbf{t}]_\times \mathbf{R}$, also called *essential matrix*, with

$$\hat{\mathbf{x}}_1^T \mathbf{E}\hat{\mathbf{x}}_0 = 0. \tag{2.14}$$

If the camera intrinsics are known the image plane coordinates $\hat{\mathbf{x}}_i = \mathbf{K}_i^{-1}\mathbf{x}_i$ can be used directly but we can also define the constraint using regular pixel coordinates $\mathbf{x}_i$

$$\mathbf{x}_1^T \mathbf{K}_1^{-T} \mathbf{E} \mathbf{K}_0^{-1} \mathbf{x}_0 = 0. \tag{2.15}$$

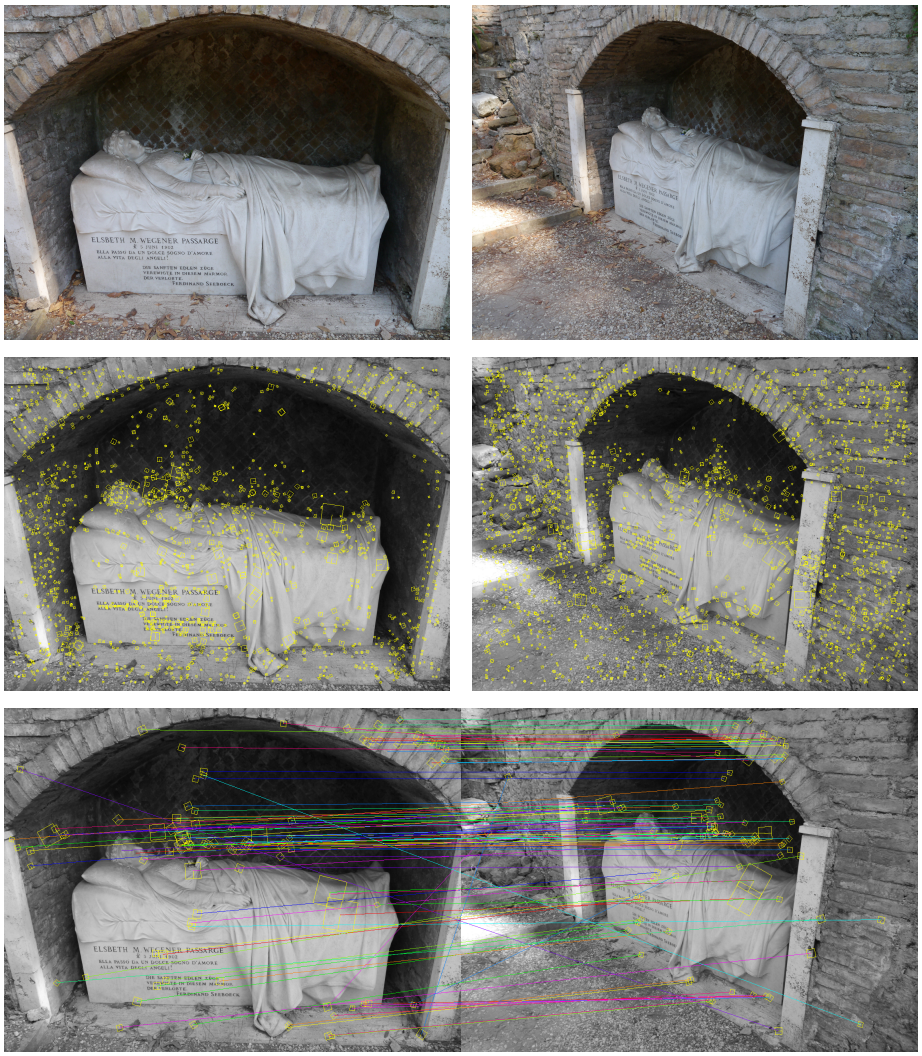


Figure 2.5: SIFT features detected in two images and the nearest neighbor matches computed from the descriptors. Note that a nearest neighbor is only selected if the distance to the second nearest neighbor is significantly high, so many features that are not very unique will not have a match in the other image.

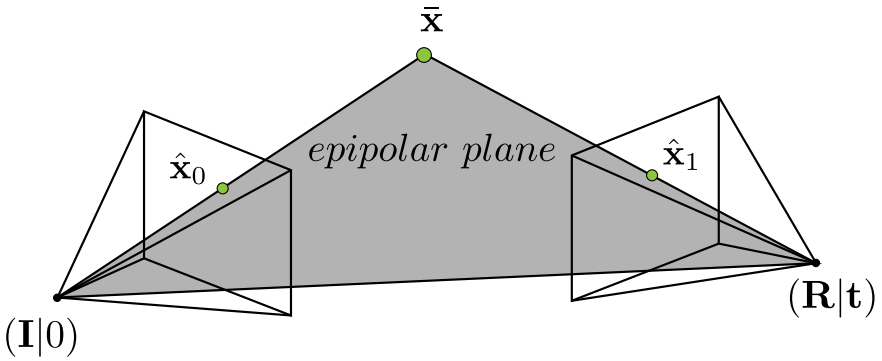


Figure 2.6: A single 3D point $\bar{\mathbf{x}}$ is projected into two cameras. The center points of the cameras and the 3D point generate the epipolar plane.

This generates a more general matrix called the *fundamental matrix* $\mathbf{F} = \mathbf{K}_1^{-T}\mathbf{E}\mathbf{K}_0^{-1}$. The fundamental matrix can be recovered from a set of at least 8 correspondences [Hartley, 1997] but it is important to handle the outlier correspondences that are returned from feature matching. The RANSAC approach [Fischler and Bolles, 1981] is the most widely used solution to this problem. It creates many, randomly sampled minimal sets of correspondences and estimates their corresponding fundamental matrix. For each matrix the distance of all other correspondences to the model is computed and for a fixed ϵ they are classified into inliers and outliers. The solution with the largest set of inliers is then selected as the best possible result. Figure 2.7 shows the filtered inlier feature matches selected by estimating a fundamental matrix within RANSAC. Once the fundamental matrix has been recovered it can be directly transformed into the essential matrix if the camera intrinsics are known. In most cases the focal length can be extracted from image meta data (EXIF tags), otherwise a guess can be provided manually. In theory it is possible to extract the focal length from the fundamental matrix [Hartley and Zisserman, 2004] but in practice the results are often inaccurate due to noisy input data. Finally the essential matrix can be separated into a relative rotation and translation of the camera pair [Hartley and Zisserman, 2004] which are then the extrinsic parameters of the second camera.

2.3.3 Triangulation

Given two cameras with known extrinsic and intrinsic parameters the 3D location of all correspondences between the two images can be determined by triangulation. The most robust solution is again derived from Equation (2.8). For every correspondence we have two pairs of pixel coordinates (x_i, y_i) that are projections of the same 3D point. This leads

Figure 2.7: *Top*: Original matches. *Bottom*: Matches filtered by removing outliers after the estimation of a fundamental matrix with RANSAC.

to a linear least squares problem for the homogeneous 3D coordinate $\bar{\mathbf{x}}$:

$$\begin{aligned} x_i = \frac{\mathbf{P}_0\bar{\mathbf{x}}}{\mathbf{P}_2\bar{\mathbf{x}}} \quad &\rightarrow \quad (x_i\mathbf{P}_2 - \mathbf{P}_0)\bar{\mathbf{x}} = 0 \\ y_i = \frac{\mathbf{P}_1\bar{\mathbf{x}}}{\mathbf{P}_2\bar{\mathbf{x}}} \quad &\rightarrow \quad (y_i\mathbf{P}_2 - \mathbf{P}_1)\bar{\mathbf{x}} = 0 \end{aligned} \tag{2.16}$$

Before constructing and solving the system the configuration of the cameras should be checked for eventual degeneracies. The uncertainty of the solution strongly depends on the angle between the two rays from the camera centers through the pixels. If the angle is too small inaccuracies in the feature location can lead to large errors in the recovered 3D location and the correspondence should be discarded.

2.3.4 Incremental Reconstruction

After the reconstruction of an initial camera pair the recovered 3D points can be used to add more cameras to the reconstruction and the incremental procedure can eventually add even more 3D points and cameras. All feature correspondences between camera pairs can be merged into larger *tracks* that connect multiple cameras. Each of these tracks has to correspond to a single 3D point. The already reconstructed points therefore also correspond

to 2D features in other uncalibrated cameras. These 3D-2D correspondences can be used to directly estimate the camera parameters. This problem is usually referred to as *Perspective-n-Point* or *PnP*, where n is the number of points used for the reconstruction. A minimal solution requires only 3 points but more can be used to increase accuracy or speed and many solutions have been developed over the years [Hartley and Zisserman, 2004; Lepetit et al., 2009; Wu, 2015]. Similar to the two view reconstruction problem this can also be affected by larger numbers of outliers so it is also necessary to use RANSAC iterations. Schönberger and Frahm [2016] also describe additional strategies to handle outlier heavy data.

2.3.5 Bundle Adjustment

As mentioned earlier it is necessary to perform a bundle adjustment every few iterations to avoid the accumulation of errors. This can be formulated as a global, non-linear optimization problem. Given a set of views V_i, 3D points p_i, and the 2D correspondences of the points in the cameras x_{ij}; the minimization error is the sum of all squared distances between the projection P of the points and the 2D features (this is also called *reprojection error*):

$$\underset{V_i,p_j}{\operatorname{argmin}} \sum_{i,j} \left(P(V_i,p_j) - x_{ij}\right)^2. \tag{2.17}$$

The projection function models all transformations of the **P** matrix and can also include non-linear effects such as radial distortion. Optimizing this objective is a more complex task, and the most common solution is to employ a Levenberg–Marquardt optimization. Triggs et al. [2000] provide an extensive overview of the problem. In recent years more focus has also been put on solving very large bundle adjustment problems fast [Agarwal et al., 2010; Wu et al., 2011b].

2.4 Stereo Reconstruction

The goal of stereo algorithms is to reconstruct 3D geometry from a set of 2D images and the corresponding calibrated camera parameters. This is achieved by recovering the depth information for each image pixel. A pixel with depth can then be interpreted directly as a sample point on a continuous 3D surface. Stereo algorithms usually formulate the search for this depth as an energy minimization problem. A given depthmap of an image is used to reproject the image pixels into a neighboring image. Starting from this reprojection an error is defined that measures how well the transformed pixels match the contents of the neighboring image. The search range of possible depth values is very large and matching image contents is a complex task. Stereo therefore quickly became a fundamental problem in computer vision [Marr and Poggio, 1976; Barnard and Fischler, 1982; Dhond and Aggarwal,

1989] and remains an active research topic. Many approaches have been developed that use different matching errors and optimization techniques. Scharstein and Szeliski [2002] provide an overview and categorization of the most successful algorithms. Most of the presented techniques only match an image to a single neighbor but it is straightforward to extend the matching error to multiple neighbors.

2.4.1 Formulation

As mentioned before stereo matching is modeled as an optimization problem. Given an image I_i from the set of views and a neighboring image I_j, the goal is to estimate a depthmap d_i for the main image depending on a cost function C that measures differences in image content. The energy is defined as

$$E_C = \sum_{\mathbf{x} \in X} C\left(I_i(\mathbf{x}), I_j(P_j(\mathbf{x}, d_i(\mathbf{x})))\right) \tag{2.18}$$

The cost function, also called matching cost or matching error, is measured pairwise between all pixels $\mathbf{x}$ and their respective projections P into the neighboring views. The projection function is easily derived from Equation (2.8): For a given depth value the projection of the main camera can be inverted and the resulting 3D point can be reprojected into the neighboring view

$$P_j(\mathbf{x}, d_i(\mathbf{x})) = \mathbf{K}_j\left(\mathbf{R}_j\mathbf{R}_i^{-1}\left(\mathbf{K}_i^{-1}\mathbf{x}_i \cdot d_i(\mathbf{x}) - \mathbf{t}_i\right) + \mathbf{t}_j\right). \tag{2.19}$$

The matching error in stereo is based on *photoconsistency*, which means that a given 3D point should have the same appearance in every image where it is seen. We know from Section 2.2 that specular materials do not necessarily conform to this assumption and therefore they cannot be reconstructed with regular stereo methods. But for diffuse reflection this assumption is generally true. To measure the photoconsistency between two locations in two images the image contents can be compared with different metrics. A single pixel does not hold enough information to construct a reliable error and most metrics use a small window around the pixel location (usually around 5×5 or 7×7 pixels). The simplest error is the sum of squared differences (SSD) of all pixels in the window. Formally for a window size of $n = k \times k$ and two intensity vectors $\mathbf{u}$ and $\mathbf{v}$ it is defined as

$$SSD(\mathbf{u}, \mathbf{v}) = \sum_{i=1}^{n} (\mathbf{u}_i - \mathbf{v}_i)^2 \tag{2.20}$$

This can be computed quickly but is not robust to changes in image or scene brightness which happens frequently for casually captured data as modern cameras adjust their exposure settings constantly. Lighting changes in outdoor scenes can also lead to more extreme and local changes in image brightness. It is therefore common to use more robust metrics. These are

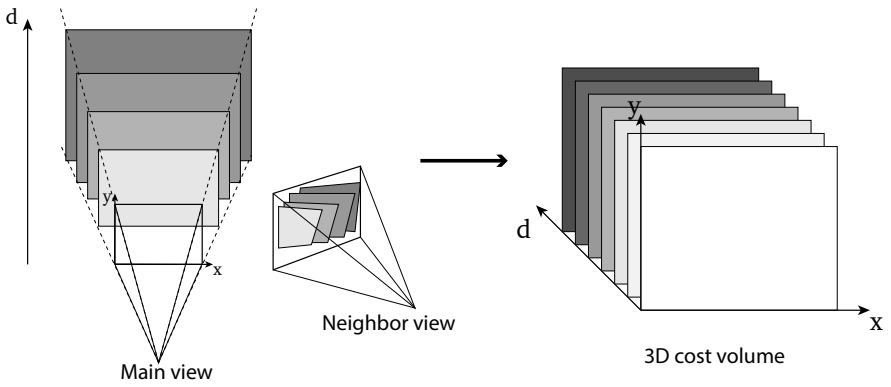


Figure 2.8: For plane sweep stereo the main view is projected into the scene for a set of discrete depth values. The plane is then reprojected into a neighboring view. Evaluating the difference between the reprojected image and the original image of the neighboring view for every pixel leads to a 3D cost volume that contains the error for every pixel and every tested depth value.

more expensive to compute but they can handle a much larger variety of input data. The most common metric is normalized cross-correlation (NCC) which first normalizes the windows by removing the mean values ($\bar{\mathbf{v}}$, $\bar{\mathbf{u}}$) from the vectors and then measures the angular difference. This is robust against linear scaling of the pixel values and therefore illumination and exposure changes. It is defined as

$$NCC(\mathbf{u},\mathbf{v}) = \frac{\sum_{i=1}^{n}((\mathbf{u}_i-\bar{\mathbf{u}})\cdot(\mathbf{v}_i-\bar{\mathbf{v}})}{\sqrt{\sum_{i=1}^{n}(\mathbf{u}_i-\bar{\mathbf{u}})^2}\cdot\sqrt{\sum_{i=1}^{n}(\mathbf{v}_i-\bar{\mathbf{v}})^2}} \tag{2.21}$$

While NCC performs well for many scenarios recent approaches also include more advanced metrics that are faster to compute or even more robust. Examples are mutual information [Hirschmüller, 2008], and census transform [Hirschmuller and Scharstein, 2009]. Hirschmuller and Scharstein [2009] also present a detailed analysis and performance comparisons for a variety of popular matching costs.

2.4.2 Optimization

Optimizing the stereo energy is very complex. The search space of all possible depth values is very large and the cost functions are non-linear and non-convex. The most widely used approach to solve this problem is to discretize the solution space, usually referred to as *plane sweep* [Collins, 1995]. For a fixed set of depth values inside a reasonable range the main image is projected into the neighboring image assuming it is a simple plane at the given depth. For each of these planes the matching cost is evaluated for all pixels leading to a

3D cost volume, see Figure 2.8. The goal is now to extract the best depth value for each pixel from this volume. A basic solution would be to select the depth with the minimum cost for every pixel but this can easily lead to large errors, especially for uniform image regions without texture where the cost function is not very discriminative. Therefore it is necessary to aggregate the matching costs. Some approaches aggregate the matching costs directly using 2D or 3D filters on the cost volume [Kanade and Okutomi, 1994; Kanade et al., 1996; Yoon and Kweon, 2006; Hosni et al., 2011]. This already improves results greatly and the filtering can be performed quickly, which makes these approaches particularly suited for real-time applications. The best results, however, are achieved by global optimization strategies. These approaches add a specific regularization or smoothness term to the cost function. These terms usually requires neighboring pixels to have similar depth values as most real surfaces are more or less smooth and depth discontinuities happen rarely. A common form for a smoothness energy E_S is

$$E_S(d) = \sum_{\mathbf{x}} \rho_1((d(\mathbf{x}) - d(\mathbf{x}+1)) \cdot \rho_2(I(\mathbf{x}) - I(\mathbf{x}+1)). \tag{2.22}$$

The first term ρ_1 now needs to be a robust loss function that can allow certain amounts of larger depth discontinuities [Black and Rangarajan, 1996]. Additionally, the second term ρ_2 is a function that vanishes for large image differences and thereby encourages depth discontinuities at large image gradients. The final optimization problem is now

$$\underset{d_i}{\operatorname{argmin}} \quad E_C(d_i) + E_S(d_i) \tag{2.23}$$

with E_C being the regular matching cost from Equation (2.18). Solving this optimization problem is based on Markov random fields and belief propagation, and for this graph cut methods have shown excellent results [Boykov et al., 2001]. The main disadvantage is that global optimization is often resource intensive, requiring a lot of memory and compute power. A detailed study comparing different approaches in this area has been published by Szeliski et al. [2008].

More recently other approaches try to operate between global and local optimization. The most prominent solution has been presented by Hirschmüller [2008]. The method searches for optimal paths in specific directions through the cost volume. These paths can be computed quickly with dynamic programming and by aggregating multiple paths from different directions a *semi-global* solution can be found for every pixel. This technique can achieve results that are only slightly worse compared to global methods, but it is much faster and can also be parallelized easily for even better performance. Another recent technique that avoids a complete global optimization is *PatchMatch* stereo [Bleyer et al., 2011; Heise et al., 2015]. After a simple random initialization this method propagates depth values that generate high photoconsistency measures to neighboring pixels. In multiple iterations this propagation is interleaved with a local refinement of depth estimates.

2.4.3 Multi-view Stereo

Multi-view stereo (MVS) generally tries to recover a complete 3D model of an object or a scene and operates on many images instead of just a single pair. It still uses the same reprojection and photoconsistency principles as regular stereo. In the simplest form the regular stereo formulation can just be extended to measure costs in many neighboring views N instead of just a single one:

$$E_C(d_i) = \sum_{j \in N} \sum_{\mathbf{x} \in X} C\left(I_i(\mathbf{x}), I_j(P_j(\mathbf{x}, d_i(\mathbf{x})))\right). \tag{2.24}$$

Goesele et al. [2006] show that this formulation with NCC as a cost function can already generate good results and is easy to implement. The general structure of this energy is used in almost every MVS algorithm. Most of the details of different techniques evolve around the representation of the geometry and the optimization techniques. Another important aspect of MVS is the scene representation. While operating on single depth maps is a widely used solution many algorithms also generate a global 3D model directly. A common approach is to use a 3D voxel representation [Seitz and Dyer, 1999]. Here the information of all images can be combined into a single optimization problem that can be solved with similar techniques as regular stereo. However, the size of this problem can quickly grow when larger voxel grids are needed to reconstruct more detail. A common way to deal with this problem is to use additional silhouette information to constrain the solution space [Furukawa and Ponce, 2009; Sinha and Pollefeys, 2005]. For uncontrolled and large datasets this can become prohibitive as silhouette information is not easily obtained. Instead of a dense voxel grid it is also possible to directly reconstruct a global 3D point cloud [Furukawa and Ponce, 2010]. This has the advantage that it can be as detailed as regular depth maps while generating a global model. It is still difficult for this representation to handle a large scene and special methods are required to separate the data into multiple smaller clusters [Furukawa et al., 2010].

Optimization techniques for MVS are very similar to two-view stereo. Starting from an optimization problem similar to Equation (2.24) it is possible to use graph cuts to find a global solution in a discretized space [Sinha and Pollefeys, 2005], to optimize single depth values with a local measure [Goesele et al., 2007], or to adapt other techniques such as PatchMatch [Galliani et al., 2015; Schönberger et al., 2016]. Another branch of optimization algorithms for MVS is based on variational refinement. Here an initial solution, often generated from the sparse output of SfM, is refined over multiple iterations. The formulation tries to minimize the matching cost between all image pairs i, j for the projection P of a continuous surface S

$$E_C(S) = \sum_{i,j} \int_S C\left(I_i(P_i(S)), I_j(P_j(S))\right). \tag{2.25}$$

To minimize this energy the integral has to be discretized which can be done globally with a triangle mesh [Hiep et al., 2009] or locally for every input image [Semerjian, 2014], leading to the same advantages or disadvantages of the global or local representations discussed earlier. While refinement algorithms can achieve high-quality results, they rely on local optimization strategies that follow the gradient of the energy function. It is therefore very important to have a good initialization.

Another important practical aspect is the selection of appropriate view pairs. Especially for very unstructured datasets — e.g. images that are acquired from different cameras, at different times, or at vastly different viewpoints — it is important to select a suitable set of images and to pay special attention to scaling and foreshortening of image content. Goesele et al. [2007] and Schönberger et al. [2016] address these issues of view selection and present algorithms that operate on difficult internet data.

Furukawa and Hernández [2015] provide a detailed look at all further challenges and recent advances in the field.

Surface Reconstruction For many rendering applications it is important to have a global triangle mesh. MVS algorithms often only provide a 3D point cloud, either reconstructed directly or a combined from all generated depth maps. A separate branch of research is focused on merging this data into a consistent surface [Fuhrmann and Goesele, 2014; Kazhdan and Hoppe, 2013; Aroudj et al., 2017; Mostegel et al., 2017], which in itself is a complex problem. Weinmann et al. [2016] provide a detailed summary of recent techniques.

Evaluation Evaluating the results of MVS is also a difficult task and specific publications deal with exactly this task [Knapitsch et al., 2017; Schöps et al., 2017; Seitz et al., 2006; Strecha et al., 2008]. To measure the quality of the reconstructed geometry the first problem is to acquire ground truth data. For this purpose many benchmarks acquire test objects and scenes with high-quality active devices such as laser scanners. During evaluation the distance of the ground truth geometry to the reconstruction is measured and used as a quality metric. This can lead to surprising results when it is compared to the pure visual quality, see Figure 2.9. Other approaches such as Szeliski [1999] and Waechter et al. [2017] therefore try to avoid these issues by directly measuring the quality of images predicted or rendered from the reconstructed geometry. This approach also makes ground truth geometry obsolete, which is especially useful for scenes where it is hard to acquire.

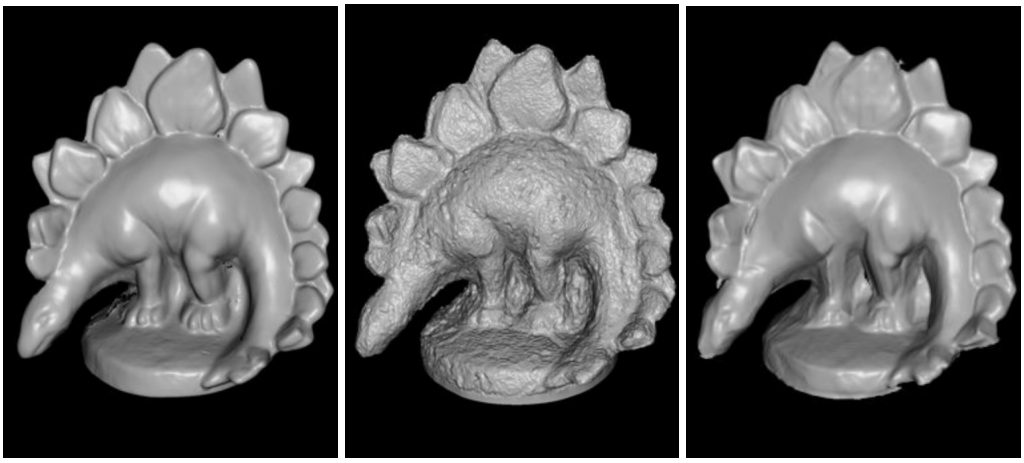


Figure 2.9: Problems with geometry based stereo benchmarks. Compared to the ground truth on the *left* the result in the *center* has the lowest geometric error in the popular Middlebury benchmark [Seitz et al., 2006]. The result on the *right* has a higher geometric error although the visual quality is much higher.

2.5 Image-based Rendering

Traditional computer graphics tries to simulate the image formation process by computing an image from scene lighting, geometry, and reflectance. As mentioned in Section 2.1 these properties can be arbitrarily complex and it is therefore very hard to achieve photorealistic results. Image-based rendering (IBR) uses a combination of graphics and computer vision based reconstructions to generate realistic experiences by modifying and combining multiple photographs. A single photo is perfectly photorealistic but it is not possible to capture an infinite amount of photos to represent a scene. IBR therefore often takes a limited set of images and renders a virtual scene using efficient interpolation and extrapolation techniques. Depending on the amount of images that are captured a variety of techniques are feasible and an approximate geometry representation can also help to reduce the amount of images required. As noted by Kang et al. [2000] there is a complete spectrum of algorithms ranging from traditional graphics with a single geometry and a single texture to full IBR rendering with a only a large set of images. Depending on the desired application and capture setup some approaches are suited more than others.

2.5.1 View Interpolation

The first IBR techniques tried to move away from traditional geometry rasterization by interpolating between a set of input images. The approach developed by Chen and Williams

[1993] is often considered the original work in the domain. Their method generates interpolated views between two input cameras using depth maps that are recovered with regular stereo methods. For a new virtual 3D camera the depth maps are used to reproject the pixels from the source images into the novel view using basic forward warping. Similar to the reprojection in stereo (Equation (2.19)) a depth value for a pixel in the source image can be used to directly map the pixel to the target image. Forward warping then simply copies the pixel of the source image to the target location. Larger holes that arise from disocclusions in one image are usually filled with content from the other image. When both images project to the same target pixel, one of the pixels can be discarded if it has a depth value that suggests occlusion, otherwise both pixels are blended if they have the same depth. This approach already works well for static scenes and good stereo reconstruction, although the forward warping often generates artifacts as the target image is not sampled regularly. Many recent approaches still build on the general idea of the original publication although the reconstruction and rendering techniques have improved significantly. Our IBR approach presented in Chapter 4 is also related to this technique.

2.5.2 View-dependent Texture Maps

Similar to view interpolation Debevec et al. [1996] also use proxy geometry to reproject input images as so called *view-dependent texture maps*. In their approach they generate a global geometry instead of a local depth map for each view. Depending on the position of the virtual camera this geometry is then textured with input images that closely match the camera's viewing angle. The texture from each image is then weighted according to the angle and all textures are blended together creating the best possible texture for the current viewing position. Although the geometry for their architectural scenes is created by hand and very coarse they can achieve very realistic results as the smooth transition between multiple input images creates an illusion of small geometric details.

2.5.3 Light field and Lumigraph rendering

At the other end of the range of IBR algorithms are light fields. The idea originated from the *plenoptic function* introduced by Adelson and Bergen [1991]. This function represents the amount of light that can be observed in every direction from every point in 3D space. This leads to a 5D space of light rays and the original formulation was also more extensive, including wavelengths and time to model more complicated scenes. Of course it is not feasible to sample this function densely as it would require a very large number of images. IBR algorithms therefore aim to reconstruct this function from a limited amount of samples. The first rendering system based on the plenoptic function was developed by McMillan and Bishop

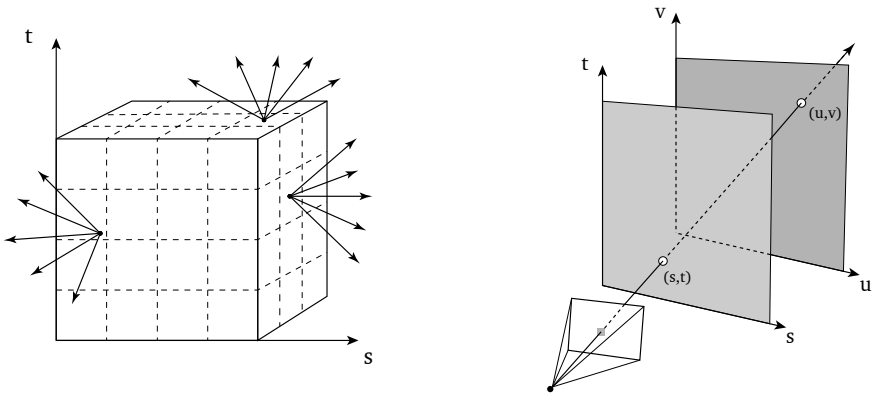


Figure 2.10: The ray parameterization used for light field and Lumigraph rendering. *Left*: The surface of the bounding volume scene is a 2D domain (a cube is usually the easiest to handle). All rays that exit this volume can be parametrized with their 2D location on the surface and their 2D direction. *Right*: This can also be thought of as a 2 plane parameterization. Every ray intersects both planes at the respective coordinates (s,t) and (u,v) which makes it easy to lookup rays that correspond to pixels from an image.

[1995]. They showed how samples of the function can be stored and accessed efficiently and how the function can be resampled to create interpolations between the original samples.

If we further assume that all light rays simply pass through the empty space of the scene and are not affected by participating media, the plenoptic function can be reduced to a 4D space of all possible light rays in the scene. This concept is usually referred to as *light field* and two very similar rendering approaches were introduced simultaneously by Levoy and Hanrahan [1996] and Gortler et al. [1996]. The approach by Levoy and Hanrahan [1996] describes a pure IBR approach without any proxy geometry while Gortler et al. [1996] also use geometry information to improve the rendering quality. Both approaches however use very similar parameterizations for the space of light rays. Assuming the visible 3D volume is bounded by a surface, for simplicity a cube, all light rays exiting this volume can be parameterized with their 2D position on the surface and their 2D direction in which they exit. Focusing on just a single side of a cube this can also be visualized by two planes st and uv. A ray in this 4D space is parameterized by its coordinates on both planes, see Figure 2.10. It is now possible to generate an image for a virtual camera outside of this volume by accessing the corresponding color values from the function $L(s,t,u,v)$. At this point Gortler et al. [1996] improve upon regular light field rendering and modify the coordinates for the lookup depending on the proxy geometry. This allows the Lumigraph to create more accurate pixel values for less dense sampled scenes.

Unstructured Lumigraph Generating a regularly sampled light field from input images usually requires complex capture setups and large amounts of data (see [Levoy and Hanrahan, 1996]). In a more general system Buehler et al. [2001] therefore describe the *unstructured Lumigraph* system which renders directly from a set of input images without explicitly reconstructing the light field. For a virtual camera the pixel values are collected and blended from the original images according to several criteria including resolution, angular deviation, and visibility. To enable real time rendering the dense blending weights for the virtual view are interpolated from sparse samples for vertices on a regular grid and additional vertices from the projected proxy geometry. Overall this technique is much more flexible than other IBR algorithms. It basically generalizes previous techniques as it approaches traditional Lumigraph rendering for scenes with densely sampled input images, but it also acts like view-dependent texture mapping for smaller amounts of images and comprehensive proxy geometry. It also allows for a much easier capturing of datasets as the capture process does not need to be very controlled and the camera positions and proxy geometry can also be reconstructed using the previously described SfM and MVS techniques.

Chapter 3

Shading-aware Multi-view Stereo

In this chapter we present our novel multi-view stereo algorithm. The main idea of this technique is to introduce shading constraints into regular geometry based stereo optimization. Previous approaches so far have struggled to do this and they relied on multiple separated steps. Our observation is that image gradients can be used to dynamically switch between traditional stereo and shading based energies. This allows us to create an optimization that is free of any regularization heuristic that was usually required in many stereo algorithms. Additionally the use of gradients also enables us to formulate our shading energy without an explicit model of the surface albedo. The results in this chapter show a clear quality improvement compared to many previous multi-view stereo algorithms.

3.1 Related Work

High-quality surface reconstruction has been an active field of research over the past decade, and approaches have been developed for various forms of input data. Our technique uses an unstructured set of images (with camera parameters) of an approximately Lambertian scene and does not require any special hardware setup. We will review related methods that either operate on similar input data or use ideas similar to our approach.

Multi-view Stereo. Multi-view stereo algorithms [Seitz et al., 2006] are arguably one of the most general passive reconstruction techniques. Approaches such as Goesele et al. [2007] and Furukawa and Ponce [2010] have shown that geometry can be recovered even for large scale and uncontrolled Internet data. Other approaches use more controlled settings or additional input such as object silhouettes [Heise et al., 2015]. Multi-view stereo approaches usually add a form of regularization to deal with structureless areas that are not well matched by classical stereo terms such as photo consistency. Similarly regularization is used in two-view stereo methods such as Hirschmüller [2005], Bleyer et al. [2011], and Galliani et al.

[2015], which can also be applied to multi-view scenarios by combining many two-view estimates into a robust multi-view estimate. In contrast, our goal is to avoid explicit regularization; instead, we use a new shading-based data term to handle sparsely textured regions where a traditional stereo term is not very effective. To do this we optimize both depth and normals of a continuous surface. In terms of surface representation, stereo algorithms usually recover a single depth per-pixel [Goesele et al., 2007], a global point cloud [Furukawa and Ponce, 2010], or an implicit surface model [Heise et al., 2015], all of which we found difficult to apply to our approach as they cannot represent a continuous smooth surface efficiently. Recently another surface representation was proposed inside a multi-view framework by Semerjian [2014]. This approach uses bicubic patches to define a surface per view that has continuous depth and normals. We found this representation to be appropriate for our method and adopt it as described later.

Combining Multi-view and Photometric Cues. To recover more detail in regions where depth reconstruction is not very accurate, several methods have combined multi-view and photometric principles. Most of them, however, rely on a controlled and complex capture setup. The approach by Nehab et al. [2005] combines two separate reconstructions. They capture depth using structured light, acquire surface normals using photometric stereo, and integrate both estimates in a separate step. Other approaches such as Hernandez Esteban et al. [2008] and Zhou et al. [2013] combine photometric stereo information from multiple view points into a single framework. This requires a large amount of input data and a complex acquisition system as both light and camera positions need to be controlled. Beeler et al. [2010] augment the geometry of captured faces with fine details using the assumption that small concavities in the skin appear darker than flat areas. They do not require a lot of input data but are still dependent on a calibrated capture setup as they do not have a variable lighting model.

Shading-based Refinement for General Illumination. Most recently, a new line of work uses shading cues from images captured under uncontrolled illumination to improve a given geometry. Wu et al. [2011c] presented the first approach that uses a precomputed multi-view stereo reconstruction to estimate a spherical harmonics approximation of the lighting. They use this lighting and a shading model to improve the stereo reconstruction. Their approach is able to recover fine-scale details but is limited to objects with a single, constant albedo. Later, Yu et al. [2013] and Han et al. [2013] both presented algorithms that operate on a single RGB-D input image (e.g., from a Kinect sensor). These sensors usually generate very coarse geometry and shading-based refinement increases the quality and resolution of the output. Xu et al. [2014] also extended the idea and developed a simultaneuos opimization of lighting and shape parameters. They do, however, require additional information about the visual hull of the object. Using GPU-based parallel solvers, Wu et al. [2014] and Or-El et al. [2015] were able to achieve real-time performance on similar input data. All these techniques are still limited to a single albedo [Han et al., 2013; Xu et al., 2014; Or-El et al., 2015], a fixed

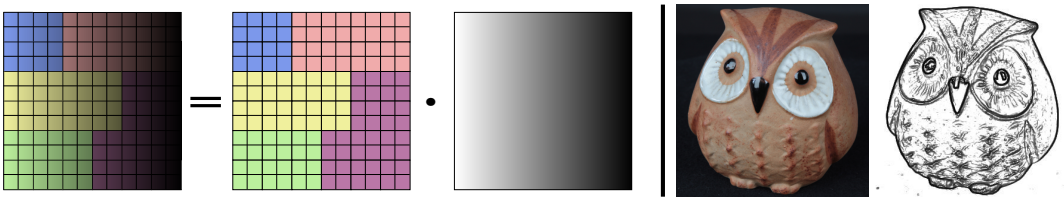


Figure 3.1: *Left*: An illustration of our Retinex-based assumption of separating albedo from shading. Large gradients in the image are usually caused by albedo changes; small gradients on the other hand are observed due to lighting. Based on this we compute a trade-off between stereo and shading energies. *Right*: Visualization of the trade-off for an input image. For every pixel we use mainly our stereo term (dark regions) or our shading term (bright regions) based on the magnitude of the image gradient.

set of constant albedo clusters [Yu et al., 2013], or a coarse initial albedo estimate [Wu et al., 2014]. Other methods focus on more specific scenarios such as faces. Chai et al. [2015] fit a parametric face model to an input image and use it for lighting estimation and shading-based refinement. The first technique to include a spatially varying albedo was proposed by Zollhöfer et al. [2015]. They include the albedo in the optimization and constrain it using a chromaticity-based regularization scheme similar to Chen and Koltun [2013]. While this prevents shading from being absorbed into the albedo, it can fail in scenes where the albedo variation is not accurately predicted by chromaticities (e.g., albedos with the same chromaticity but different brightness).

Although shading-based refinement techniques have improved significantly in recent years, the basic principle of all existing methods remains the same: They use fixed input geometry, estimate lighting, and later refine the geometry using shading cues. While we also compute a lighting function on a coarse estimate of the geometry, we integrate the geometry refinement directly into the multi-view stereo reconstruction method. This allows us to balance stereo matching and shading cues as we can resolve ambiguities in the multi-view stereo energy, instead of treating the input geometry as fixed. This approach ultimately also enables us to optimize the geometry independently of the (potentially spatially-varying) albedo, i.e., without explicitly including albedo terms into our energy. This is a significant advantage because we do not need to rely on albedo regularization models that can often fail on real-world scenes.

3.2 Framework

Our energy balances geometric errors versus shading errors depending on the local image gradient. This is motivated by Land's Retinex theory [Land, 1977], which assumes that

shading introduces only small image gradients, changing the surface brightness gradually. Strong gradients on the other hand are usually caused by changes in surface materials and are thus independent of the illumination. Retinex theory has been commonly used to separate surface albedo and shading [Horn, 1974; Grosse et al., 2009] (see Figure 3.1).

In our context, this observation has two implications. First, in multi-view reconstruction the geometric stereo term is usually accurate and robust in regions with strong gradients but fails for small gradients. Many stereo methods therefore use surface regularization to keep textureless areas smooth. We instead utilize the fact that small gradients are most likely caused by lighting and define an additional data term based on a shading function that specifically constrains the direction in which the surface should change. Second, we show that, in regions of small gradients, we can factor the surface albedo out completely, resulting in an albedo-free shading term. Our error terms are based purely on point wise image gradients and do not involve image values or larger patches of pixels.

The input to our algorithm is an unstructured set of images as well as known camera parameters which can be either pre-calibrated or recovered by stucture from motion tools such as VisualSFM [Wu et al., 2011a]. We aim to compute a depth map for every view i using a set of neighbor views $j \in \mathcal{N}_i$.

3.2.1 Geometric Error

Our camera model follows standard definitions [Hartley and Zisserman, 2004]. A 3D point $\mathbf{X}$ is transformed into an image location $\mathbf{x}$ in the camera coordinate system according to a camera calibration matrix $\mathbf{K}$, rotation $\mathbf{R}$, and translation $\mathbf{t}$ as

$$\mathbf{x} = \mathbf{K}(\mathbf{R}\mathbf{X} + \mathbf{t}). \tag{3.1}$$

For homogeneous coordinates the projection from a pixel coordinate $\mathbf{x}_i$ in camera i into another camera j can then be defined according to a depth value $d_i(\mathbf{x}_i)$ along the principal ray of view i:

$$P_j(\mathbf{x}_i, d_i(\mathbf{x}_i)) = \mathbf{K}_j \left(\mathbf{R}_j \mathbf{R}_i^{-1} \left(\mathbf{K}_i^{-1} \mathbf{x}_i \cdot d_i(\mathbf{x}_i) - \mathbf{t}_i \right) + \mathbf{t}_j \right) \tag{3.2}$$

The geometric error is now defined as a stereo term based on matching intensity gradients from the main view into neighboring views according to the current depth function. Traditional stereo methods often optimize using image values over a local patch of pixels. Even for illumination invariant measures such as normalized cross-correlation, this would be more difficult to integrate into our Retinex assumption as a patch of pixels is more likely to be affected by both albedo and shading changes. Instead, we specifically optimize this energy for local image gradients. A gradient-based stereo term was introduced by Scharstein [1994] but was never widely adopted in a multi-view setting. Semerjian [2014] recently showed

that a point-wise measure of gradients can be very effective for surface reconstruction if used correctly. We adopt this measure as it is well suited for our approach. For any two views i, j and their intensity functions I_i, I_j, and a pixel coordinate $\mathbf{x}_i$ it can be written as:

$$E_g^j(d_i, \mathbf{x}_i) = \nabla I_i(\mathbf{x}) - \nabla I_j(P_j(\mathbf{x}_i, d_i(\mathbf{x}_i))). \tag{3.3}$$

Here, and in further equations, ∇ denotes image gradients which are the derivatives computed with respect to image coordinates $\mathbf{x}_i$. Note that this also involves the derivative of the projection P_j which transforms the gradient into the correct coordinate system. In addition to constraints beween the main view and its neighbors, we also define pairwise terms between two neighbors as used by Semerjian [Semerjian, 2014]. Still using the depth of the main view d_i we get:

$$E_g^{j,k}(d_i, \mathbf{x}_i) = E_g^j(d_i, \mathbf{x}_i) - E_g^k(d_i, \mathbf{x}_i) = \nabla I_j(P_j(\mathbf{x}_i, d_i(\mathbf{x}_i)) - \nabla I_k(P_k(\mathbf{x}_i, d_i(\mathbf{x}_i)), \tag{3.4}$$

where $E_g^{i,j} = E_g^j$. This essentially measures the difference in error between neighbors and avoids overfitting to only one neighbor.

3.2.2 Shading Error

Lighting Model: Similar to previous work [Wu et al., 2011c][Zollhöfer et al., 2015] we assume Lambertian reflectance. This allows us to define shading as a function of the surface normal $\mathbf{n}$, independent of the viewing direction. We use third-order spherical harmonics basis functions B_h to approximate the incoming illumination. The outgoing radiance $R(\mathbf{x})$ at a point $\mathbf{x}$, with albedo $a(\mathbf{x})$ and normal $\mathbf{n}(\mathbf{x})$, is a weighted sum of these bases, which we define as our shading function S:

$$R(\mathbf{x}) = a(\mathbf{x}) \cdot \sum_{h=1}^{16} B_h(\mathbf{n}(\mathbf{x})) \cdot \mathbf{l}_h = a(\mathbf{x}) \cdot S(\mathbf{n}(\mathbf{x}), \mathbf{l}) \tag{3.5}$$

The lighting parameters $\mathbf{l}$ are computed ahead of surface optimization using a coarse initial surface model derived from basic stereo. This optimization is identical to Zollhöfer et al. [2015], i.e., we initialize the albedo as constant and simply solve a linear least squares system. In contrast to Zollhoefer et al., we optimize $\mathbf{l}$ using only our single main image. Using more images would make this estimation more robust, but we explicitly want to optimize for a separate lighting model per image to be invariant to changing light conditions, e.g., an object moving on a turn table or outdoor scenes with uncontrolled lighting. We also set our albedo to a constant value. As we will describe later, we are able to optimize the geometry without explicitly modeling the albedo. This has many advantages for the optimization procedure, but unlike Zollhoefer et al. we cannot create an improved lighting model in further iterations. While there are obvious scenarios that will break this approach, the low number

of lighting parameters causes the estimation to be robust enough for a variety of objects, as we will demonstrate in the results. In fact, we observed that in practical scenarios it is much more likely that errors appear due to specular surfaces, self shadowing and inter-reflections, which cannot be dealt with in either case.

Shading Error: Our shading term is also based on image gradients. Similar to Zollhöfer et al. [2015], we assume that the observed image gradient, ∇I, should be identical to the gradient of the reflected intensity predicted by our model, ∇R, with:

$$\nabla R(\mathbf{x}) = \nabla a(\mathbf{x}) \cdot S(\mathbf{n}(\mathbf{x}), \mathbf{l}) + a(\mathbf{x}) \cdot \nabla S(\mathbf{n}(\mathbf{x}), \mathbf{l}). \tag{3.6}$$

However, at this point we do not have an accurate model of the albedo. Previous approaches therefore include the albedo in the optimization leading to a significantly bigger, under-constrained problem. This requires an explicit regularization on the albedo using approximate measures such as pairwise differences based on chromaticity. Instead, we use the Retinex assumption to create an albedo independent optimization that does not require any explicit regularization. A common approach for intrinsic images [Horn, 1974; Chen and Koltun, 2013] is to operate in the log domain as this makes albedo and shading terms additive instead of multiplicative:

$$\log(R(\mathbf{x})) = \log(a(\mathbf{x})) + \log(S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})). \tag{3.7}$$

If we take the gradient with respect to image coordinates we get:

$$\nabla \log(R(\mathbf{x})) = \frac{\nabla a(\mathbf{x})}{a(\mathbf{x})} + \frac{\nabla S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})}{S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})}. \tag{3.8}$$

If we now assume—according to the Retinex theory—that small gradients are caused solely by lighting, the albedo gradient vanishes and we can write:

$$\nabla \log(R(\mathbf{x})) = \frac{\nabla S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})}{S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})}. \tag{3.9}$$

This means that the difference, $\nabla \log(I(\mathbf{x})) - \nabla \log(R(\mathbf{x}))$, can in fact be minimized by solely optimizing over the shading function, $S(\mathbf{n}(\mathbf{x}), \mathbf{l})$. This indicates an albedo invariance which can also be thought of in the following way: If the albedo is locally constant, an intensity gradient is only caused by a change in surface normals, and given a lighting model, the surface normals have to change in a particular direction which does not depend on the actual value of the albedo. Our shading error is therefore defined as

$$E_s(d_i, \mathbf{x}) = \frac{\nabla I(\mathbf{x})}{I(\mathbf{x})} - \frac{\nabla S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})}{S(\mathbf{n}(d_i(\mathbf{x})), \mathbf{l})}. \tag{3.10}$$

Note that this is a simple point-wise measure which matches the point-wise nature of our gradient-based stereo term and suggests a balanced optimization if both are combined.

3.2.3 Combined Energy

To formulate our final energy function we combine both data terms in a simple but effective way. For pixels with strong gradients, we rely on the geometric stereo term as it is very robust. For small gradients, we additionally use our shading error as it constrains the surface according to the given lighting model. As we want to do this on a per-pixel basis, we need a continuous trade-off to avoid artifacts. Our solution is to use the magnitude of the image gradient to compute a weight on the shading error term, see Figure 3.1 for an example. For a set of neighbors, $\mathcal{N}_i$, including i itself, and a set of pixels, $\mathcal{V}_i$, that are visible in the corresponding neighbors, the final energy is defined as:

$$E(d_i) = \sum_{j,k \in \mathcal{N}_i}^{k>j} \sum_{\mathbf{x}_v \in \mathcal{V}_i} |E_g^{j,k}(d_i, \mathbf{x}_v)| + \frac{\alpha}{\|\nabla I(\mathbf{x}_v)\|_2} |E_s(d_i, \mathbf{x}_v)|, \tag{3.11}$$

where $\alpha = 0.01$ balances the scale of both terms as the shading error is measured in the log domain. We use the same value for all our datasets. We also experimented with normalizing the weight across pixels. The new weight β would then also affect the geometric error, i.e., $(1-\beta)E_g + \beta E_s$, resulting in a total weight of 1 for each pixel. However, this led to worse results. Note that the final energy is constructed only with local measures and does not contain any explicit regularization terms. Instead it is implicitly regularized by the Retinex assumption and the lighting model. We also use the L1 norm for both our data terms as it is more robust to outliers that do not correspond to our Retinex assumption. It also avoids scale issues in the optimization that can be caused by the shading energy becoming very large in dark areas.

3.3 Energy Minimization

As discussed earlier, we use the framework of Semerjian [2014] to optimize our energy function. It provides a surface representation with a continuous definition of depth values and surface normals which is very beneficial for our combined energy. Optimizing a depth map for each view allows us to handle datasets with varying lighting conditions and enables straight forward parallel processing. As this framework uses a different approach compared to simple pixel-wise depth values, we briefly summarize the main aspects.

3.3.1 Surface Representation

The surface is not represented as depth values per pixel but rather as a set of bicubic surface patches. Every patch is defined by bicubic interpolation between 4 nodes, and neighboring patches share two nodes (see Figure 3.2). A node itself represents 4 optimization variables:

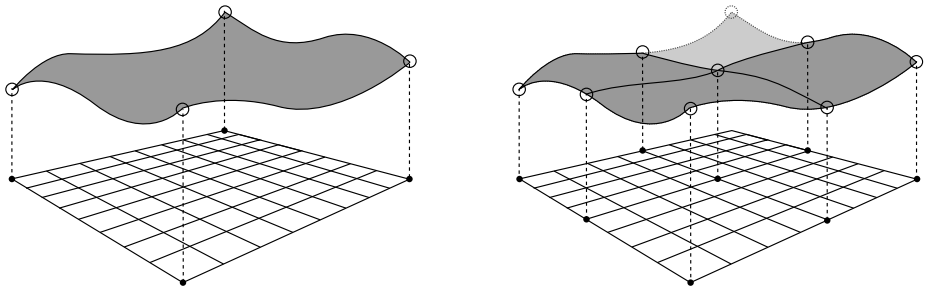


Figure 3.2: Surface representation based on bicubic patches. Each patch is defined via 4 nodes (illustrated as circles) that are located at pixel corners (illustrated as dots on the pixel grid). When moving to a higher scale the patch is subdivided and some patches are removed if they have a high error.

the depth, the first derivatives of the depth and the mixed second derivative. The nodes are located at image coordinates of the main view and each bicubic patch covers a set of pixels. This also enables an easy formulation of scale, as patches can cover more pixels to represent a coarser scale and can be subdivived to move to a finer scale. At the finest scale the patches cover a 2×2 set of pixels.

3.3.2 Optimization

Given this representation, we can efficiently optimize the non-linear energy (Eqn. 3.11) using a Gauss-Newton type solver. As our shading error is albedo-free, we do not need to introduce additional variables and can operate solely on the surface representation. This is important as it leads to an efficient structure of the linear system that is solved in each step. Starting from an initial guess the current energy is linearized, and we solve for an update to the optimization variables. Let $\mathbf{d}$ be the vector of optimization parameters, $\hat{\mathbf{d}}$ the update, and $\mathbf{f}(\mathbf{d})$ the vector of residuals generated by our energy E. Linearizing the error function around the current solution using the Jacobian, $\mathrm{J}_{\mathbf{f}}$, leads to the common linear system:

$$\mathbf{f}(\mathbf{d}+\hat{\mathbf{d}}) \approx \mathbf{f}(\mathbf{d}) + \mathrm{J}_{\mathbf{f}}^T \hat{\mathbf{d}}, \qquad \left(\mathrm{J}^T \mathrm{J}\right)\hat{\mathbf{d}} = -\mathrm{J}^T \mathbf{f} \tag{3.12}$$

Note that the Jacobian can become very large for even small image sizes. For n neighbors every pixel generates $\frac{n!}{2!\cdot(n-2)!}$ errors and therefore rows in the matrix. It is therefore more efficient to directly compute the entries of the approximate Hessian $\mathrm{H} = \mathrm{J}^T\mathrm{J}$. This matrix consists of 4×4 blocks that correspond to the 4 optimization variables at each node. The outer product of every row of the Jacobian generates a linear part for 4 of these blocks, corresponding to the 4 nodes of the patch that generated the depth and error for the respective pixel. We therefore directly compute these outer products for every error and add the blocks to the Hessian. The final matrix H is also very sparse due to the limited support of the bicubic

patches; each node is used for a maximum of 4 patches. Furthermore $J^T J$ is automatically symmetric so in our implementation we only compute and store the upper triangular part of the matrix. Note that every pixel can be handled independently and computing the error does not require high bandwidth memory access. This suggests that an efficient massively parallel implementation should be possible, we did however not pursue that idea for this work.

The final linear system is sparse, symmetric, and positive definite, and can be solved efficiently using a standard conjugate gradient solver. The inverse of the block diagonal of $J^T J$ is also a good preconditioner and can be computed quickly using Cholesky decompositions on the blocks.

3.3.3 Final Algorithm

The non-linear optimization described in the previous section can only converge to a local optimum. It is therefore important to have a good initial approximation. Semerjian [2014] suggests using the sparse reconstruction from SfM to create an initial surface. We found that this often leads to larger holes and other artifacts when parts of the scene do not have very descriptive image features and are therefore not reconstructed well by SfM. In our approach we therefore first run a plane sweep stereo with semi-global matching [Hirschmüller, 2005] on a downsampled resolution of the image. This can be done quickly and it generates a more accurate and more dense initialization than simply reprojecting sparse SfM points. The overhead in computation time is usually also recovered later as the non-linear Gauss-Newton optimization requires fewer iterations to converge.

Starting from this initial geometry we first run a few iterations of the multi-scale surface operations of Semerjian [2014] for coarse scales. Smaller scales with patch sizes of 8×8 pixels and lower are then optimized using our new energy. Applying our shading term for coarse scales would not improve the final result as geometry details are only revealed at finer scales. Another reason is efficiency; the shading error additionally involves the gradient of the shading function and is therefore more complicated to compute which increases the runtime compared to simple regularization. An example for the optimization on different scales is shown in Figure 3.3.

Finally, the reconstructed surfaces from all views are converted to a point set with normals and can be fused with any surface reconstruction algorithm [Kazhdan and Hoppe, 2013; Fuhrmann and Goesele, 2014; Ummenhofer and Brox, 2015; Aroudj et al., 2017]. Each view can also be represented as a depth or normal map.

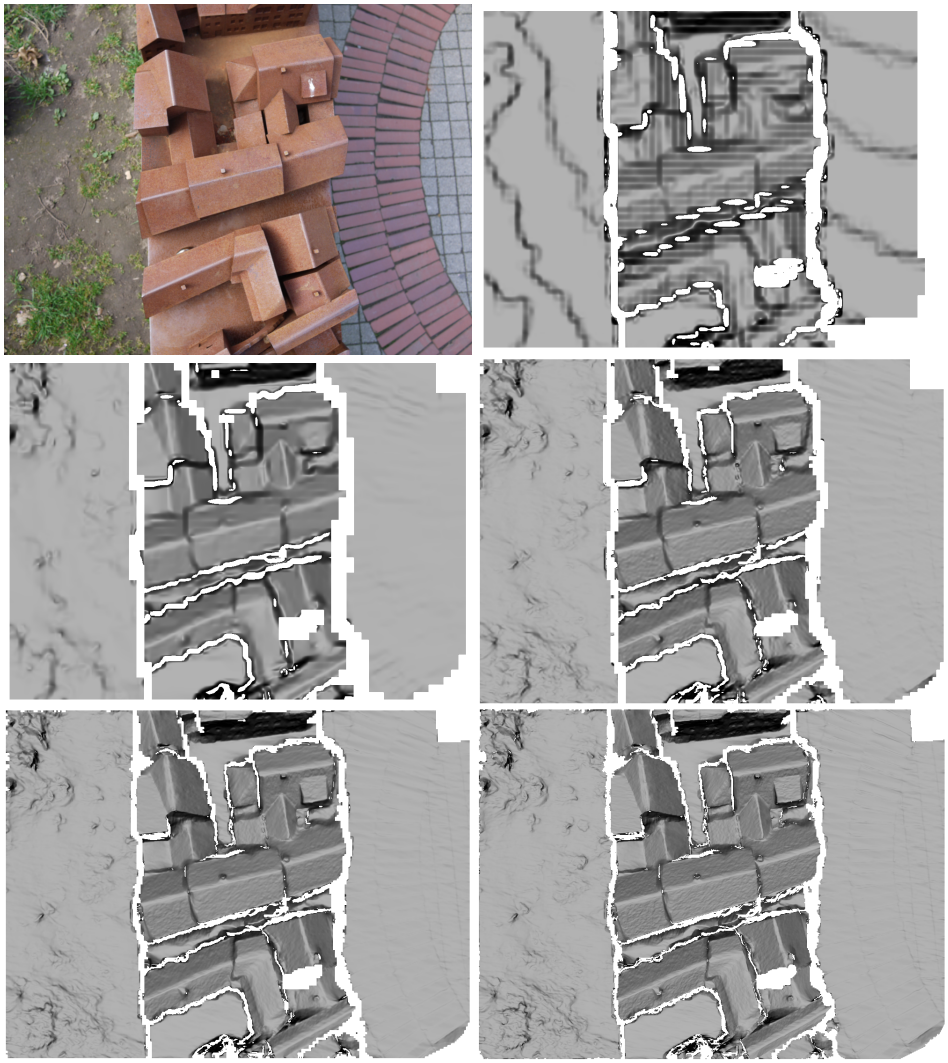


Figure 3.3: Multi-scale optimization for shrinking patch sizes. *Top*: Input image and initial depth map (rendered with Lambertian shading) computed with semi-global matching [Hirschmüller, 2005]. *Middle*: Optimization on a coarse scale using basic surface regularization and the next finer scale using our shading-based energy. *Bottom*: Optimization on the finest scales (patch sizes 4×4 and 2×2 pixels) using our shading-based energy. The difference gained on the smallest scale can be marginal and this step could also be skipped if runtime is a concern.

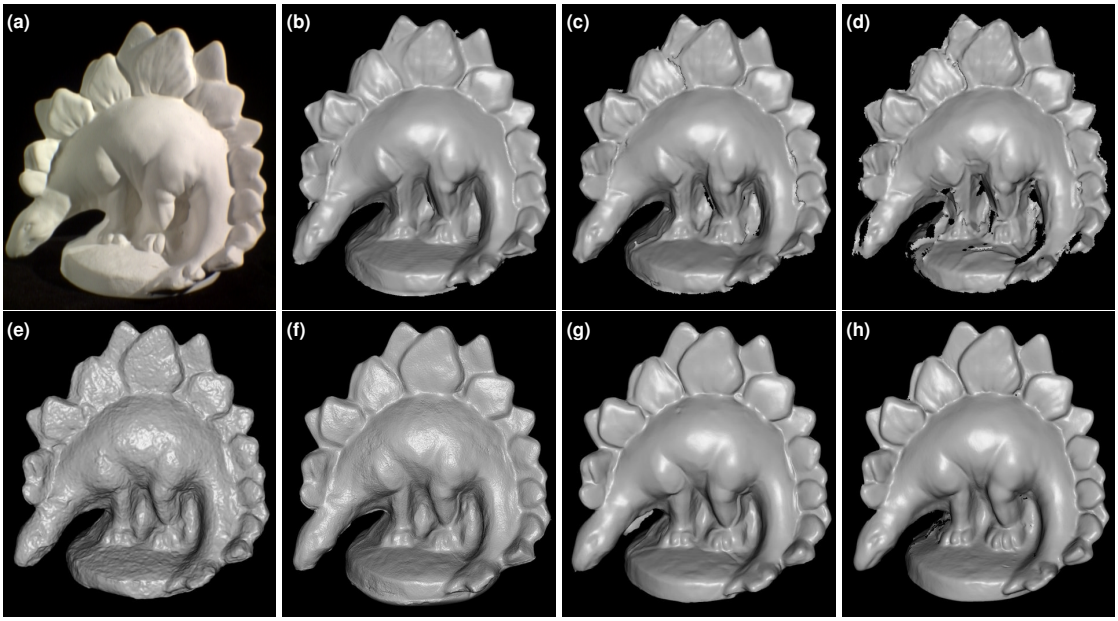


Figure 3.4: Results on the *Dino* dataset of the Middlebury benchmark with decreasing number of input images. This dataset has strong shadowing which can be seen in the input image. However, in areas where our lighting model is correct we are able to recover a high amount of detail in the geometry even for sparse input data. *Top*: (a) Input image; (b) our reconstruction on full dataset, 363 images, using 9 neighbors; (c) ring dataset, 46 images, using 4 neighbors; and (d) sparse ring dataset, 16 images, using only 1 or 2 neighbors. *Bottom*: Results on full dataset submitted by (e) Furukawa and Ponce [2010], (f) Galliani et al. [2015], and (g)Semerjian [2014]; and (h) ground truth.

3.4 Results

In the following, we evaluate our method using a variety of datasets. For all our results we used 6-9 neighbor images (except for the sparse Middlebury datasets) and fused them into a global model using Floating Scale Surface Reconstruction (FSSR) [Fuhrmann and Goesele, 2014]. We chose this approach because it does not fill holes that may appear in the geometry due to large errors in our stereo and/or shading energy.

We first evaluate our approach on the well known Middlebury benchmark [Seitz et al., 2006]. Comprehensive results are available on the website. The *Dino* dataset has many areas that are affected by self shadowing and interreflections. As Figure 3.4 shows, our optimization can handle these effects in many cases if enough stereo information from multiple views is available. Note that our optimization handles cast shadows to some extent implicitly since the weight for the shading term is low at the shadow boundaries, and cast shadows can be matched well with stereo matching. The lighting model is, however, still wrong

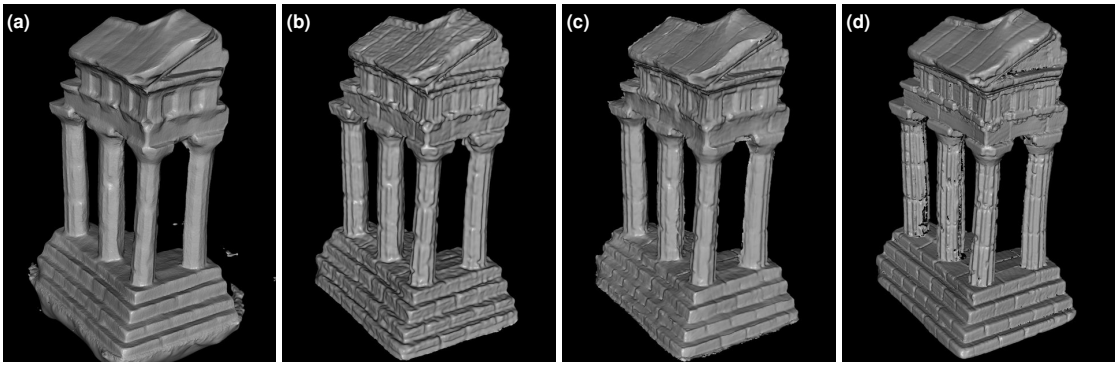


Figure 3.5: Results on the *Temple* dataset of the Middlebury benchmark. *From left to right*: (a) [Galliani et al., 2015]; (b) [Fuhrmann and Goesele, 2014] using the stereo from [Goesele et al., 2007]; (c) our reconstruction; and (d) ground truth. Our reconstruction achieves a good balance between capturing fine-scale detail without introducing noise.

Algorithm	**Temple Full (Acc. - Comp.)**	**Dino Full (Acc. - Comp.)**
Furukawa and Ponce [2010]	0.49mm - 99.6%	0.33mm - 99.8%
Galliani et al. [2015]	0.39mm - 99.2%	0.31mm - 99.9%
Semerjian [2014]	0.62mm - 97.8%	0.39mm - 99.9%
Ours	0.47mm - 98.7%	0.49mm - 96.9%

Table 3.1: Comparision of quantitative Middlebury evaluations on the full datasets (363 images).

inside the shadowed areas since the incoming illumination is partially occluded. On the full dataset our result has an accuracy of 0.49mm and a completeness of 96.9%. For the sparse *Dino* dataset where stereo cues are not very strong, our shading term causes holes in the shadowed areas as we cannot find consistent normals in these areas. However, compared to other approaches, we are able to recover a significant amount of detail in areas that are not affected by shadows. In fact, we reconstruct the same amount of detail independent of the sparsity of the input data, which highlights another strength of our shading term. Even for the very sparse input data of 16 images and using only 2 neighbors we can reconstruct more detail than top scoring approaches on the full dataset. For the full *Temple* dataset (Figure 3.5), we are able to achieve a high accuracy even though the back of the object has many concavities leading to strong interreflections that cannot be represented by our global lighting model. Compared to the results submitted by Semerjian [2014] our shading term improves the accuracy on the full dataset by 0.15 mm to 0.47mm and we achieve a completeness of 98.7%. Table 3.1 shows the full quantitative comparison. More data is available on the benchmark website.

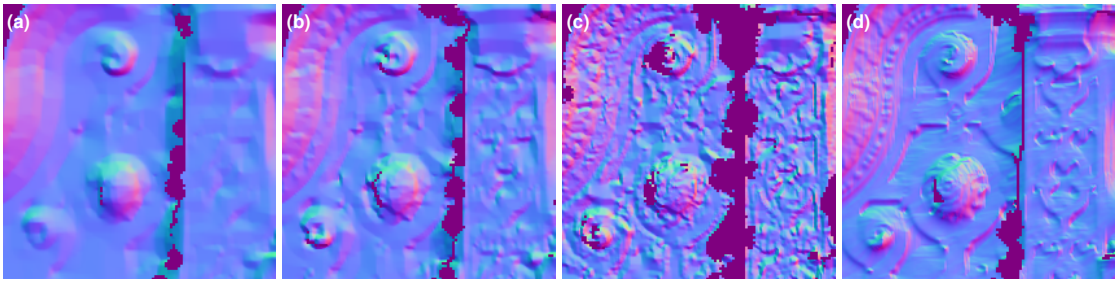


Figure 3.6: The *fountain-P11* dataset from Strecha et al. [2008]. *From left to right*: Closeup normal maps for single views of the bottom left area for different weights on surface regularization (a) high, (b) medium, and (c) low; and (d) normal map of our reconstruction. Basic regularization cannot find a good trade-off between overly smooth and noisy geometry. Our result reveals fine details without introducing noise.

Figure 3.7: The *fountain-P11* dataset from Strecha et al. [2008]. *From left to right*: Reconstruction by our implementation of Semerjian [2014] using a low regularization weight to recover details; by our new optimization; and ground truth.

Figures 3.6 – Figure 3.9 show *fountain-P11*, an outdoor dataset from the Strecha et al. [2008] benchmark. The normal maps in Figure 3.6 and Figure 3.8 show the effect of different surface regularization weights on the original approach of Semerjian [2014]. There is no globally correct weight as the reconstructed geometry is either too smooth or too noisy. In contrast, our approach reconstructs smooth but detailed geometry due to the image gradient magnitude-based weight. Figure 3.7 demonstrates that this also translates to the fused geometry as integrating multiple views cannot remove the noise inherent in Semerjian's reconstruction effectively. Figure 3.9 compares our reconstruction against the method by Goesele et al. [2007]. Note that the center figure in this dataset is also not Lambertian which violates the assumptions of our shading term and can lead to errors. Finally, Figure 3.10 shows another comparison that highlights the quality of our depth maps on another dataset from the benchmark.

Figure 3.8: Comparison against basic surface regularization as used by Semerjian [2014]. (a) Input image. (b) Normals of our reconstructed depth map for the input image. (c, d, e) Normal of depth maps reconstructed with our implementation of the basic surface regularizaion by Semerjian [2014] for various weights on surface regularization (strong, medium, weak).

Next we present a multi-scale outdoor dataset included in the FSSR paper [Fuhrmann and Goesele, 2014]. Figure 3.11 shows that our approach can recover detailed geometry in such a setting. The normal map captures even the finest details recovered in a single view. Our results from vastly different scales can be combined into a consistent model with FSSR. However, we can observe the boundaries between scales as the resolution and accuracy of the geometry changes drastically. This still illustrates an advantage compared to other systems that operate on a global model: our approach can scale to any amount of images and can easily reconstruct different levels of detail in a single dataset, whereas keeping a multi-scale global model in an efficient data structure is challenging and not arbitrarily scalable.

Figure 3.12 shows a dataset presented by Zollhöfer et al. [2015]. This object already provides many gradients for stereo matching so we do not expect our shading term to result in a substantial improvement. Note, however, that our reconstruction has significantly better quality compared to the normal map reconstructed with Semerjian's approach, and compared to the Zollhöfer et al. [2015] reconstruction provided on their project web page. Fig-

Figure 3.9: Comparison against Goesele et al. [2007]. *Left*: Reconstruction using Goesele et al. [2007] and FSSR [Fuhrmann and Goesele, 2014] *Middle*: Our reconstruction after fusing depth maps with FSSR. *Right*: Ground truth geometry.

ure 3.13 presents an additional result provided on the site and we can again observe a clear quality improvement in our reconstruction.

Finally, Figure 3.14 presents results on a dataset captured under varying lighting conditions. The *Owl* was captured on a turn-table with fixed lights and a fixed camera, resulting in different lighting for each image (w.r.t. the image coordinates). The object is nearly diffuse apart from the dark specular areas where all the methods shown here fail. We compare against a patch based stereo method [Goesele et al., 2007], which has no effective regularization as each pixel is optimized independently. This results in a very uneven surface and noise in (almost) textureless regions. Semerjian [2014] uses a simple regularization term that keeps the surface variation low. This is effective in producing a continuous surface, but cannot recover details in regions without strong gradients. In contrast, our combined method recovers a smooth surface and is able to relate small gradients to surface details. Figure 3.15 shows two additional views and the corresponding depth maps generated by our method. Note that the depth maps have been reconstructed for two vastly different viewpoints to the sides of the object. We can observe errors for regions around specular highlights but we still recover a consistent model when viewed from the front.

3.4.1 Runtime

A C++ implementation of our technique is available as open source software [1]. This unoptimized prototype shows a roughly 20 % runtime increase compared to our implementation of Semerjian [2014]. In practice, the full *Dino* and *Temple* datasets were computed in 75 and 63 minutes on a 32-core machine using Intel Xeon E5-2698 processors. The multi-scale

[1] https://github.com/flanggut/smvs

outdoor dataset from Fuhrmann et al. [Fuhrmann and Goesele, 2014] included 204 high resolution images and was computed in 115 minutes on the same machine, while the *Owl* dataset with 10 images took around 7 minutes. For a fair comparison to other stereo methods, we are reporting the run-times of our complete multi-view algorithm and not only the time required for solving our shading-based optimization.

3.4.2 Limitations

We make two main assumptions in our method that can lead to errors in the final geometry if they are violated. First, we assume that the scene is Lambertian and a low frequency spherical harmonics lighting can accurately represent the illumination. As we show in the Middlebury *Dino* dataset, shadows and interreflections will cause errors in the reconstruction but we are still able to reconstruct details in areas where our lighting model is correct. A more sophisticated lighting model could solve the issues in future work, and would require only minor changes to our geometry optimization. Second, we assume that we can separate albedo and lighting according to the magnitude of the image gradient. While this holds for many datasets, there are objects where the albedo changes gradually, and this violation of Retinex can show up in our geometry if we relate these small gradients to shading and therefore changes in the surface normal. This suggests that some geometry regularization might still be needed in certain regions where we cannot easily decide between albedo and shading. As we rely solely on the stereo error for strong gradients, we are also limited by its accuracy. In certain configurations, e.g., observing horizontal lines under horizontal camera motion, or fine structures with aliasing effects, the stereo term might lead to wrong depth estimates that we cannot fix with our normal-based shading term.

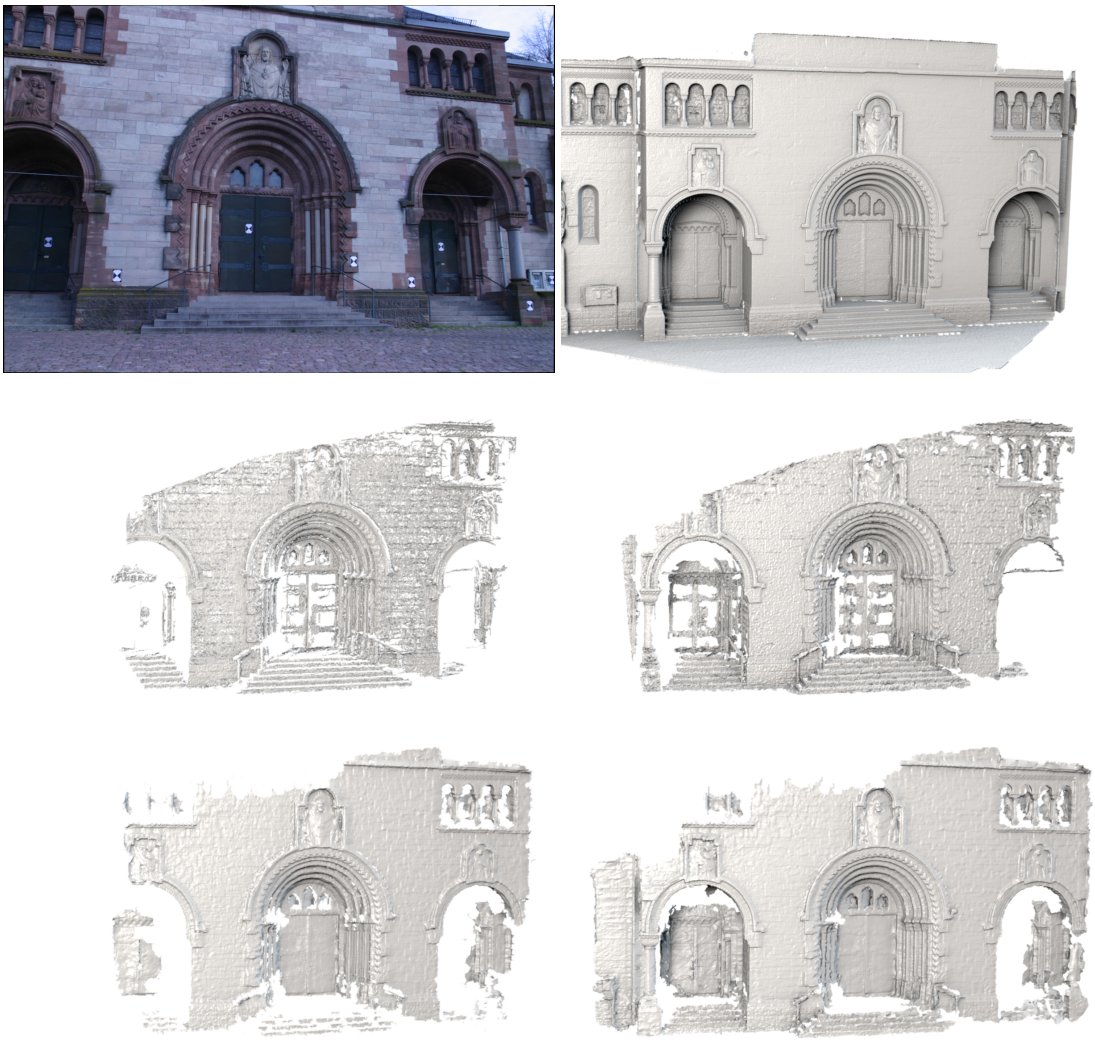


Figure 3.10: Another comparison against Goesele et al. [2007] on the *herz-jesu-p8* from Strecha et al. [2008] dataset with 8 images. *Top*: Input image and ground truth geometry. *Middle*: Reconstructed depth map using Goesele et al. [2007] and model fused from all 8 images using FSSR [Fuhrmann and Goesele, 2014]. *Bottom*: Our reconstructed depth map and reconstruction after fusing depth maps with FSSR. Our algoritm shows a more complete reconstruction with detailed geometry and less noise.

Figure 3.11: Results on an outdoor dataset. *Top*: Input images at different scales, and our global model with details. Our method recovers more detail in regions that are imaged at higher-resolution. *Bottom*: A closeup input image; the reconstructed depth map shaded with the lighting; and close-up normals with regular (10^{-2}) and low (10^{-4}) value for α – decreasing the weight of the shading term results in more noise and less detail as the stereo term dominates the energy.

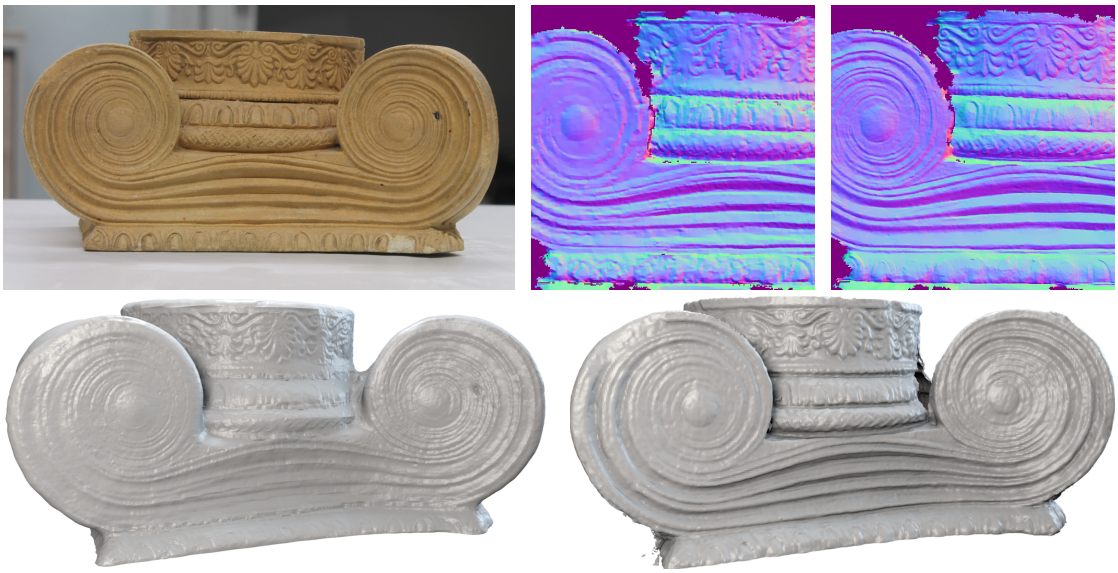


Figure 3.12: The *Figure* dataset. *Top from left to right*: An input image of the dataset; normal maps from the surface computed by our implementation of Semerjian [2014], and our shading based approach. *Bottom*: Result presented by Zollhöfer et al. [2015] (available at project website); and our fused model.

Figure 3.13: Comparison against the *Vase* dataset from Zollhöfer et al. [2015]. *Left*: Input image. *Middle*: Reconstruction by Zollhoefer et al. (available on project page). *Right*: Our reconstruction.

Figure 3.14: Reconstruction of the *Owl* dataset with changing lighting in each image. *From left to right*: An input image; reconstruction by Goesele et al. [2007]; by our implementation of Semerjian [2014]; and using our new optimization. Our results capture more structural details (see the eyes for example) with less overall noise.

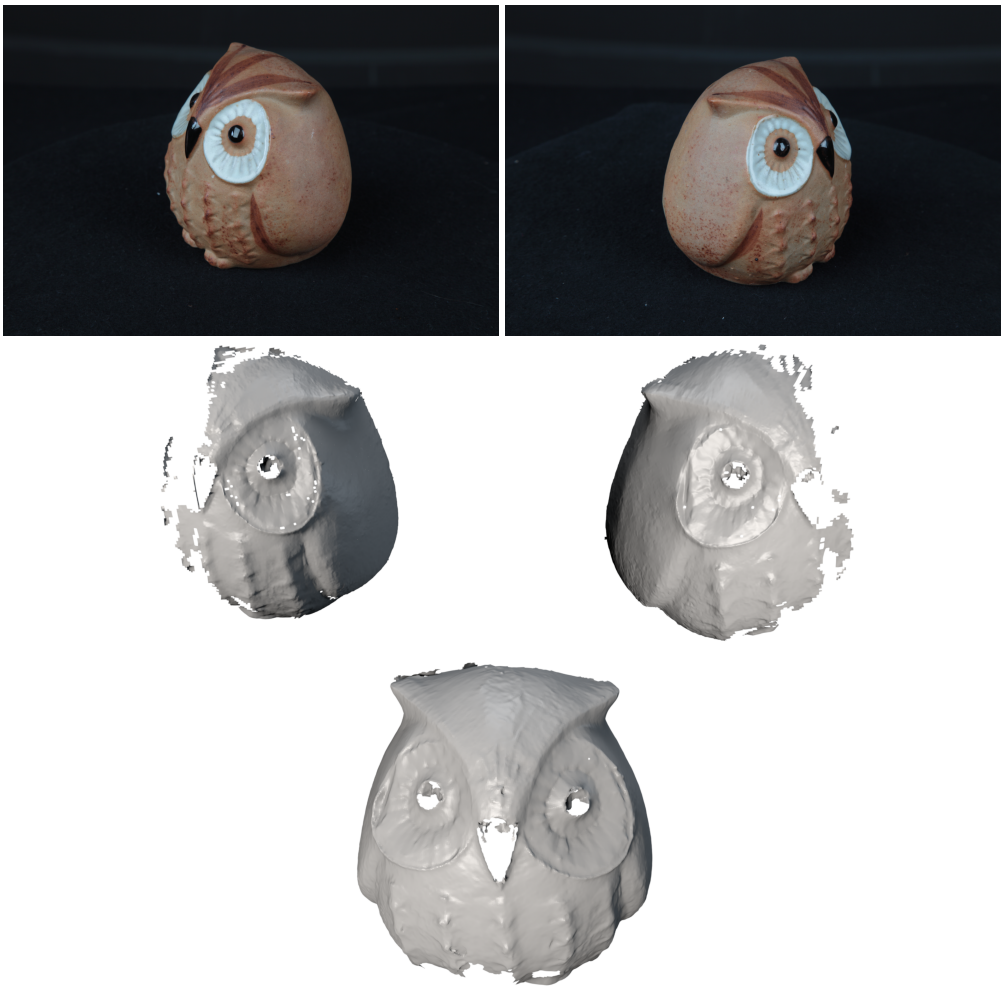


Figure 3.15: Additional results on the *Owl* dataset. *Top*: Two additional input images. As the object was captured on a turntable with fixed lights and fixed camera, leading to a different illumination on the object for every image. The specular reflections reveal the light direction to be always behind the camera. *Middle*: Depth maps recovered by our algorithm for the images shown above. *Bottom*: Fused reconstruction using only the two depth maps shown above.

Chapter 4

Gradient Domain Rendering

This chapter presents our gradient domain algorithm for image-based rendering. The key idea is related to the stereo method from Chapter 3: Stereo generally works well in the presence of strong image gradients. Starting from this observation our rendering method explicitly generates images from gradients instead of pixel values. We therefore explicitly focus on the areas where we know stereo produces accurate results. Areas with small gradients do not contribute much to our final rendering and we can therefore hide reconstruction errors. Our results show that this has advantages compared to previous methods as it generally handles textureless and reflective surfaces much better. We present a detailed explanation of how gradients can be reprojected into novel views and how the final image can then be obtained via integration. Our method also explicitly handles occlusions which is more complicated compared to pixel based rendering as gradients with different depth don't necessarily occlude each other.

4.1 Related work

Image-based rendering takes pre-rendered or captured images of 3D scenes and interpolates these images to create novel in-between views [Shum et al., 2007]. While image-based rendering can be performed purely in ray space without the need for any 3D proxy geometry [Levoy and Hanrahan, 1996], more accurate results (for the same amount of data) can be obtained by mapping input views onto some estimated proxy geometry and then blending between adjacent views [Chen and Williams, 1993; Debevec et al., 1996; Gortler et al., 1996; Buehler et al., 2001]. Over the years, a wide variety of algorithms and representations have been developed to recover and model such geometry, including global polyhedral models [Debevec et al., 1996; Gortler et al., 1996; Buehler et al., 2001] and piecewise planar "impostors" [Shade et al., 1998; Popescu et al., 2006; Sinha et al., 2009].

Several recent approaches are concerned with handling scenes where accurate depth estimation is challenging. Eisemann et al. [2008] present a technique to correct misaligned projections on coarse 3D proxies. Goesele et al. [2010] turn uncertain pixels into randomized "ambient point clouds", effectively replacing reconstruction errors by less objectionable noise. Chaurasia et al. [2013] hallucinate plausible depth in poorly reconstructed regions based on appearance similarity to well reconstructed regions. Lastly Penner and Zhang [2017] explicitly turn uncertainties in the reconstruction into a volumetric geometry model from which they render new images in ray marching type approach. This approach can generate very good results but is too slow for real-time rendering.

Even with 3D geometry, the movement of visual features in scenes that contain both reflected and transmitted light cannot be correctly modeled, since two different motions can be present at such locations. (Reflections also do not obey rigid *epipolar geometry* when the reflective surfaces are curved or undulating [Criminisi et al., 2005].) A lot of research has been done in recovering such *transparent motions* in computer vision using both layered motion models [Shizawa and Mase, 1991; Bergen et al., 1992; Irani et al., 1994; Ju et al., 1996; Szeliski et al., 2000] as well as more complex models that can handle multiple reflections or use frequency-domain analysis [Diamant and Schechner, 2008; Beery and Yeredor, 2008]. The separation and modeling of specular highlights has also received a lot of attention [Bhat and Nayar, 1998; Carceroni and Kutulakos, 2002], as has the separation of reflections using polarizing filters [Schechner et al., 1999], focus [Schechner et al., 2000] and the analysis of transparency in single images [Levin et al., 2004].

Relatively less work, however, has focused on recovering transparent and reflected motions for the purpose of image-based rendering. Szeliski et al. [2000] demonstrate how to model a scene with local planar depth approximations and to then separate the colors of the transmitted and reflected light using constrained least squares. Tsin et al. [2006] recover general depth maps corresponding to the transmitted and reflected light using a graph-cut optimization framework that estimates up to two depths per pixel. Most recently, Sinha et al. [2012] introduce a two-stage approach that uses semi-global stereo matching [Hirschmüller, 2008] followed by a piecewise-planar approximation to model complex scenes with reflections and gloss. After estimating the contribution of transmitted and reflected light in each image, they develop a real-time image-based rendering system that blends between the original images using this additive two-layer decomposition. While the results they demonstrate often work well, their approach sometimes produces visual artifacts near curved surfaces and in areas where either the scene reflectivity and/or the 3D geometry is inaccurately estimated. Most often, these artifacts are visible in the separated layers, which can have "ghosts" corresponding to the other layer or ringing due to errors in the least-squares fitting stage (Figure 4.10).

In this work, we sidestep the need to estimate two complete proxy geometries and to separate

each input image into transmitted and reflected light. Instead, we concentrate on getting good motion (depth) and occlusion estimates at strong gradients in the image, and then use Poisson reconstruction to synthesize each new frame from the motion of the displaced gradients.

Several recent image interpolation techniques synthesize results in the gradient domain. Mahajan et al. [2009] present a technique that determines optimal linear paths in image space between pairs of pixels. They use gradients as part of the matching cost and to reduce artifacts during rendering. The interpolated frames are recovered using a 3D space-time Poisson reconstruction, where the images to be interpolated are used as boundary conditions. Linz et al. [2010] present a related approach where interpolated views are generated by warping gradient images based on a dense flow estimate followed by a similar Poisson integration. Our algorithm differs from these approaches in that it more deeply exploits the fundamental properties of gradients such as sparsity and separation of reflections: the ability to move individual gradients enables handling reflections and other non-Lambertian effects. Our system also uses a full 3D scene model and can render the scene from arbitrary viewpoints. We do not use the input images as boundary conditions since we are not restricted to interpolating images, but instead propose a novel data term to regularize the 2D Poisson reconstruction.

4.2 Overview

Given a set of input photos, our goal is to synthesize images from novel viewpoints by performing image-based rendering in the gradient domain. In traditional image-based rendering, the 3D scene is modeled as (potentially coarse) geometric proxies and then used to reproject the input images into a novel view. In this work, instead, we consider how the image of a scene changes if we move the viewpoint and interpret this as movement of the image gradients (i.e., the edges between pixels) in the rendered image. In order to achieve physically correct movement, we assign 3D positions to the gradients, which enables us to render them from any viewpoint.

It is straightforward to assign 3D positions to surface texture gradients. If the gradient is caused by a scene discontinuity (occlusion boundary), however, it is less obvious what 3D position it should be assigned. In this case we assign the gradient the depth of the occluder and achieve (at least approximately) the correct behavior. Finally, there are gradients on semi-transparent reflective surfaces, e.g., a sheet of glass with the underlying scene shining through. This case is very challenging for traditional reconstruction methods. However, the gradients from the two layers are typically still well separated, due to the property that gradients in natural images are sparse. For this reason it is unlikely that edges from the two layers coincide; we will most likely observe no (strong) gradient at a pixel, or only a single

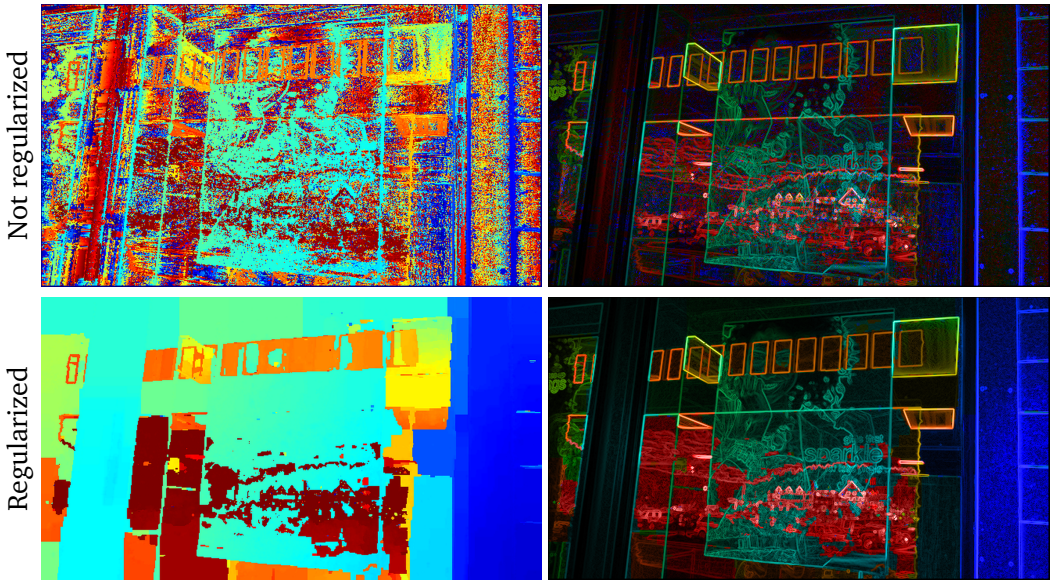


Figure 4.1: Stereo reconstruction, unregularized (top) and regularized (bottom). In the right column we modulated the depth maps by gradient magnitude to visualize that the depth is correct at image edges. Regularization produces more consistent results, while still being correct at strong gradients.

(strong) gradient. See Figure 4.1 for an illustration. This property has been used previously to separate reflections in a single image [Levin et al., 2004].

4.2.1 Scene reconstruction

The first step in most image-based rendering systems is estimating the camera parameters as well as (dense) depth maps for the input images. We use a standard approach for this stage in our system. In particular, we use a structure from motion pipeline similar to the one described by Snavely et al. [2006] to estimate the location, orientation, and intrinsic parameters for the input cameras. Next, we run plane sweep-based multi-view stereo matching with normalized cross-correlation (NCC) as the photometric consistency measure and combine it with graph cuts [Kolmogorov and Zabih, 2002] to extract a dense depth map for each input view. In all of our experiments, we use 256 discrete depth labels.

Given two horizontally or vertically neighboring pixels p_1 and p_2, we compute pairwise

regularization weights

$$w_{12} = \begin{cases} 0 & d_1 = d_2, \\ 0.005 & \|d_1 - d_2\| = 1, \\ 0.200 & \|d_1 - d_2\| > 1, \ \|\mathbf{g}_1\| < \epsilon, \ \|\mathbf{g}_2\| < \epsilon, \\ c_\angle & \|d_1 - d_2\| > 1, \ \|\mathbf{g}_1\| > \epsilon, \ \|\mathbf{g}_2\| > \epsilon, \\ 0.005 & \text{else (high and low gradient magnitude)}, \end{cases} \tag{4.1}$$

where d_1, d_2 are the depth labels, and $\mathbf{g}_1, \mathbf{g}_2$ are the color gradients at the pixel positions. $c_\angle = \mathbf{g}_1^\top \mathbf{g}_2 / \|\mathbf{g}_1\| \|\mathbf{g}_2\|$ is the dot product of the normalized gradient vectors. We used $\epsilon = 0.075$ in all our experiments. This formula favors consistent depth between gradients with low magnitude (as they most likely belong to the same surface) and between gradients with a higher magnitude and similar direction (since they likely belong to the same edge). Depth discontinuities are preferred between two gradients with a high and a low magnitude.

In contrast to traditional approaches, we only need reliable depth estimates at pixels with non-negligible gradient magnitudes. Fortunately, these are just the locations where any vision-based 3D reconstruction method works best. Figure 4.1 shows the reconstruction results with and without regularization. In the right column, we modulate the depth map by the gradient magnitude to show that the depth is correct at image edges (despite regularization that favors piecewise planar surfaces). In other words, the depth is correct in places where it matters for our method, whereas wrongly estimated depths away from image edges will affect traditional image-domain rendering methods, where they lead to artifacts.

4.2.2 Rendering

Our method reconstructs the novel view first in the gradient domain, by computing two gradient fields F_x, F_y, for horizontal and vertical gradients, respectively, and then obtains the final color image I through integration. Note that whenever we talk about gradients in this chapter, we are referring to forward differences, i.e., the gradients can be thought to be localized on the right and bottom edges of pixels for horizontal and vertical gradients, respectively (see Figure 4.3).

Computing the interpolated gradient field is easy: we start with an empty image and simply splat the gradients where they project to in the novel view using additive blending. The challenge lies in the integration step, which recovers the color image. Due to noise or missing data, the gradient fields are generally not fully integrable. Thus, we obtain the color image with the best matching gradients by solving a Poisson problem [Pérez et al., 2003]. However, this solution is only defined up to an arbitrary additive constant. Unlike previous work [Pérez et al., 2003; Mahajan et al., 2009; Linz et al., 2010] we do not have a fixed boundary and

thus cannot use boundary conditions to obtain a non-singular linear system. Instead we regularize the solution with a weakly weighted data term.

We tried several simple data terms, such as a constant color or the depth reprojected input image (Figures 4.5a–b). However, our particular application gives us a better option (Figure 4.5d). Since we start with regular images (the inputs) and render the scene from a different but similar perspective, we can create the result images using a pure image-domain rendering approach. More specifically, we observe that shifting a gradient in an image by a number of pixels will simply change the value of the pixels over which the gradient passed by the gradient magnitude, which is either added or subtracted depending on the direction of the movement. This can be implemented in a simple rendering step (see Sections 4.3 and 4.4 for details). In a perfect world where the 3D reconstruction would yield accurate and noiseless results, this would theoretically give perfect results. However, since vision is not perfect, this solution contains artifacts. It still suffices, however, as an approximate solution (or data term), S, which we can use to weakly bias the additive offset of the Poisson problem.

Overall, we solve the following optimization problem:

$$\min_I \left(\tfrac{\partial}{\partial x} I - F_x\right)^2 + \left(\tfrac{\partial}{\partial y} I - F_y\right)^2 + \lambda\left(I - S\right)^2, \tag{4.2}$$

where $\lambda = 0.1$ is used to weakly bias the solution towards our approximation. We implemented a simple solver using the Conjugate Gradient Method [Shewchuk, 1994] in CUDA that runs in real time on the GPU.

As with other view interpolation systems, we render a novel view in our system always from *two* reference images. For this, we find the respective closest input camera to the left and to the right of the novel view. We then compute the term images F_x, F_y, S separately for both inputs and combine each corresponding pair of terms into a single term image using linear blending weights proportional to the inverse distance of the cameras. Finally, we solve a single combined Poisson system using Equation 4.2.

4.3 Horizontal camera motion

In this section, we describe a simple special case that illustrates the key concepts of our method while avoiding some of the complications that arise in the general case. In this setting, the camera moves horizontally (basically corresponding to a rectified setting). In this case, the epipolar lines are parallel to the x-axis and all points move only horizontally between the original and novel viewpoints, without any scaling. This makes splatting the gradient fields F_x and F_y particularly easy: we distribute each gradient's value to the two nearest pixels using linear splat kernels.

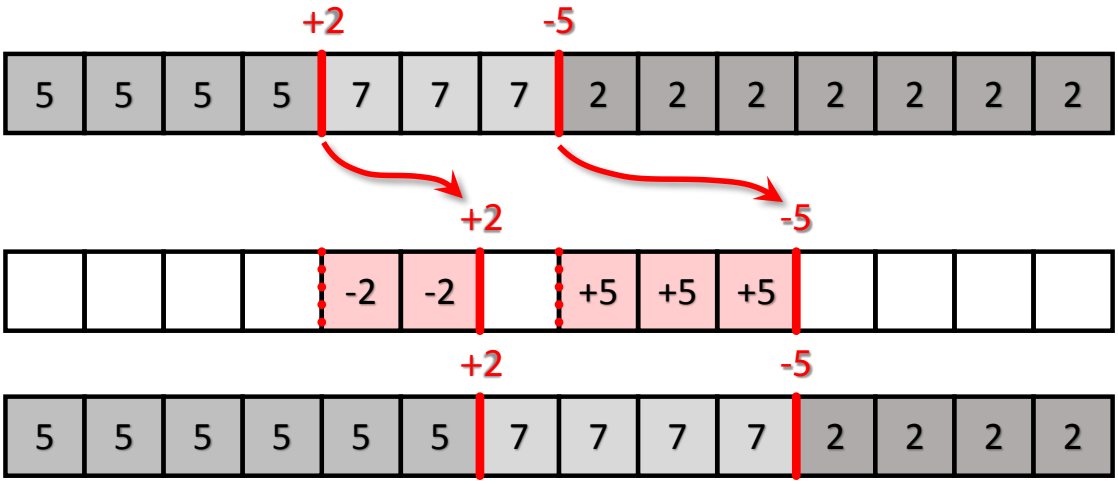


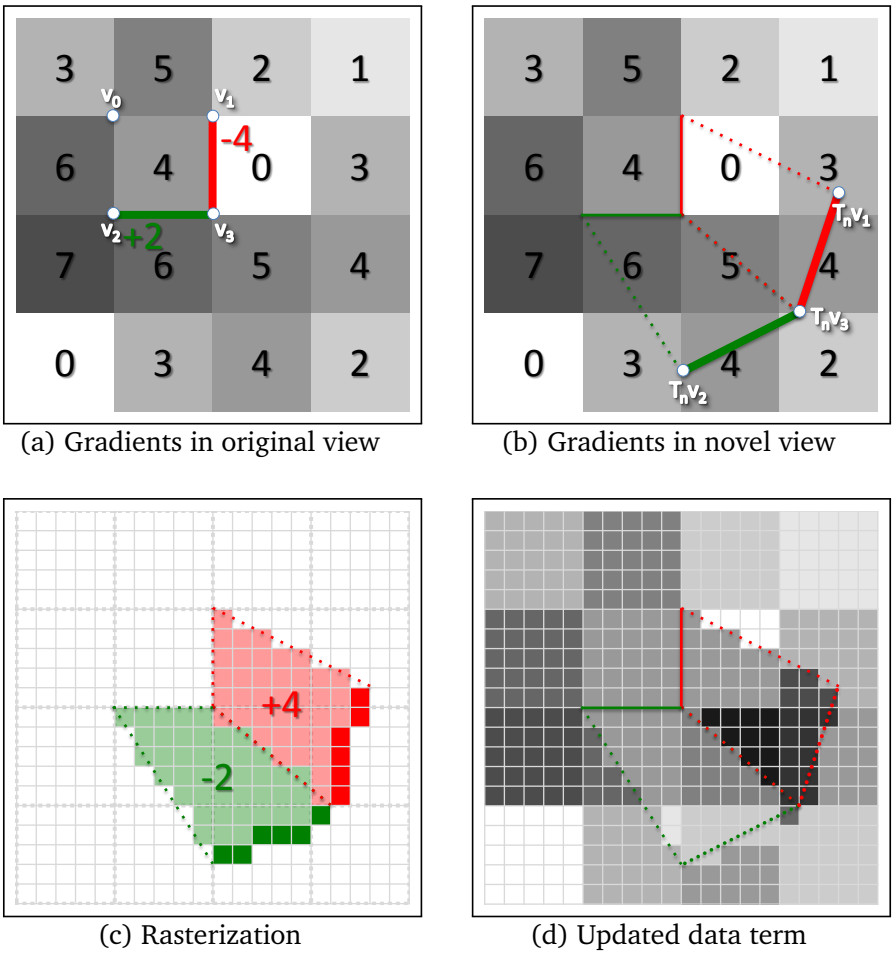
Figure 4.2: Computing the approximate solution in the special case of pure horizontal camera motion. The top row shows a scanline from the source image with the two non-zero gradients highlighted. The gradients shift to their new locations (on the same scanline because of the restricted camera motion model). As gradients shift right across the image, their value is *subtracted* from the underlying pixels (bottom rows).

To compute the approximate solution S, we initialize it by copying the input photo and then shift the horizontal gradients one by one from their original locations to their new locations, updating S as the gradients move across each pixel (Figure 4.2). For this operation, we consider the gradients to be located on the boundary between pixels (red lines in Figure 4.2). A gradient that shifts right is *subtracted* from the image, and a gradient that shifts left is *added*. This operation is only applied to the horizontal gradients in this section. The vertical gradients do not affect the approximate solution S. In practice, we implement this operation by rendering a one pixel wide horizontal line for each gradient, connecting the original and new locations, using the appropriate color value and additive blending.

Given F_x, F_y, and S we can now solve the Poisson problem given in Equation 4.2 using the conjugate gradient solver.

4.4 General camera motion

Handling general camera motion involves the same steps as the horizontal case described before: splatting the gradient fields F_x, F_y, and computing an approximate solution S. However, it also poses some new challenges that we did not encounter before. First of all, gradients can now move both horizontally and vertically. Moreover, the mapping from original



(a) Gradients in original view (b) Gradients in novel view

(c) Rasterization (d) Updated data term

Figure 4.3: Computing the approximate solution in the general case. Each gradient (a) generates a quad connecting its locations in the original and novel view (b). The quad is rasterized (c) and splatted with additive blending updating the underlying image (d). In Figures (c) and (d), the small pixels can also be thought of as the super-samples used in anti-aliasing. The dark red and green lines in (c) are the spine of the 1-pixel wide anti-aliased lines used to splat the gradients into the gradient buffer.

to novel view can involve rotation and scaling (of pixels) that we have to handle.

Let us define some notation first. We consider the pixels in the original image as fronto-parallel squares in 3D world space, whose corner vertices $\mathbf{v}_0, \ldots, \mathbf{v}_3$ are located at the depth estimated as described in Section 4.2. The horizontal gradient is assumed to be located at the right boundary, $\overline{\mathbf{v}_1\mathbf{v}_3}$, and the vertical gradient at the bottom boundary, $\overline{\mathbf{v}_2\mathbf{v}_3}$, colored red and green in Figure 4.3, respectively.

Let $\mathbf{T}_\mathrm{o} = \mathbf{P}\mathbf{M}_\mathrm{o}$ and $\mathbf{T}_\mathrm{n} = \mathbf{P}\mathbf{M}_\mathrm{n}$ be the transformation matrices that map homogeneous world vectors to screen space for the original and novel view, respectively. $\mathbf{P}$ is a 4×3 projection matrix, and $\mathbf{M}_\mathrm{o}, \mathbf{M}_\mathrm{n}$ are 4×4 model-view matrices, all defined for a right handed coordinate system (e.g., as in the OpenGL API).

4.4.1 Splatting gradients

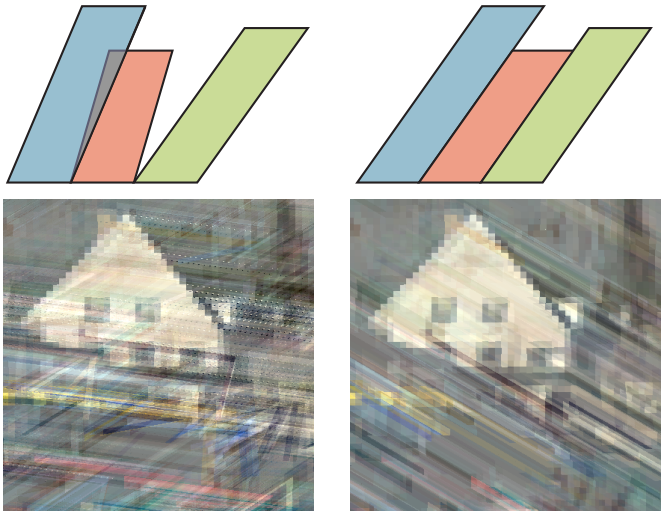
When generating the gradient term images, we need to splat line segments rather than points, to account for zoomed and rotated gradient boundaries in the novel view. For each horizontal and vertical gradient, we rasterize a 1 pixel-thick line segment connecting $\mathbf{T}_\mathrm{n}\mathbf{v}_i$ and $\mathbf{T}_\mathrm{n}\mathbf{v}_j$, where i and j index the two corners involved. The gradient values are splatted in an additive manner as before. Figure 4.3c shows the spines of these anti-aliased lines as dark red and green lines.

4.4.2 Computing the approximate solution

The goal of computing the approximate solution remains the same as before, but we now have to consider both horizontal and vertical gradients. We first compute the exact continuous geometry of the region affected by a gradient by forming a quad Q connecting the two pixel corners in the original and novel views, i.e., $Q = (\mathbf{T}_\mathrm{o}\mathbf{v}_i, \mathbf{T}_\mathrm{o}\mathbf{v}_j, \mathbf{T}_\mathrm{n}\mathbf{v}_j, \mathbf{T}_\mathrm{n}\mathbf{v}_i)$. In Figure 4.3b we show the resulting geometry for a pixel's horizontal and vertical gradients. The quads are rasterized to determine the pixels involved (Figure 4.3c). For a horizontal/vertical gradient that moves right/down, its value is *subtracted* from the frame buffer, otherwise it is *added* (Figure 4.3d). A simple way to implement this condition is to test whether the quad is front- or back-facing.

4.4.3 Rectified streaks

A complication when computing the approximate solution in the general case is that gradients are not all moving in the same direction anymore. This can lead to artifacts, e.g., gaps opening between neighboring gradients and clutter due to gradients overlapping chaotically



Not rectified Rectified

Figure 4.4: Rectification removes gap artifacts and clutter from the approximate solution.

(see Figure 4.4). We correct this behavior by rectifying the original and novel cameras, so that all gradient shift directions become aligned. (Alternatively, we could use a more sophisticated model for the pixel geometry, deviating from the fronto-parallel assumption.)

Rectifying two cameras involves rotating them such that the epipolar lines are aligned with the x-axis [Loop and Zhang, 1999]. We achieve this by replacing the original model-view matrices with new ones,

$$\mathbf{M}_\mathrm{o} := \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t}_\mathrm{o} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \quad \mathbf{M}_\mathrm{n} := \begin{bmatrix} \mathbf{R} & -\mathbf{R}\mathbf{t}_\mathrm{n} \\ \mathbf{0}^\top & 1 \end{bmatrix}, \tag{4.3}$$

where $\mathbf{t}_\mathrm{o}$ and $\mathbf{t}_\mathrm{n}$ are the positions of the original and novel cameras, and $\mathbf{R} = \begin{bmatrix} \mathbf{r}_\mathrm{right} & \mathbf{r}_\mathrm{up} & -\mathbf{r}_\mathrm{front} \end{bmatrix}^\top$ is the usual rotation matrix.

The camera right vector is set parallel to $\overline{\mathbf{t}_\mathrm{o}\mathbf{t}_\mathrm{n}}$,

$$\mathbf{r}_\mathrm{right} = \frac{\mathbf{t}_\mathrm{n} - \mathbf{t}_\mathrm{o}}{\|\mathbf{t}_\mathrm{n} - \mathbf{t}_\mathrm{o}\|}, \tag{4.4}$$

and the two other base vectors are obtained through cross products,

$$\mathbf{r}_\mathrm{front} = \frac{\mathbf{w}_\mathrm{up} \times \mathbf{r}_\mathrm{right}}{\left\|\mathbf{w}_\mathrm{up} \times \mathbf{r}_\mathrm{right}\right\|}, \text{ and} \tag{4.5}$$

$$\mathbf{r}_\mathrm{up} = \mathbf{r}_\mathrm{right} \times \mathbf{r}_\mathrm{front}. \tag{4.6}$$

Here, $\mathbf{w}_\mathrm{up}$ is the global world space up vector.

There are some issues with this approach. First, the rotation changes the novel view, so we would have to use image warping in a post-process to get the desired result. Warping is undesirable because it can introduce sampling artifacts. More critically, the rectified views can become extremely distorted under certain conditions, in particular, if one camera center approaches the frustum of the other.

We remedy both problems by computing an extra screen space homography H that maps the rectified novel view back to the *non-rectified* novel view, and applying this homography to both novel and original geometry *before* rasterization, i.e., $\mathrm{H} = \mathbf{T}_\mathrm{o}\mathbf{T}_\mathrm{n}^{+} = \mathbf{T}_\mathrm{o}\mathbf{T}_\mathrm{n}^{\top}\left(\mathbf{T}_\mathrm{n}\mathbf{T}_\mathrm{n}^{\top}\right)^{-1}$. The final transformation matrices are therefore $\mathbf{T}'_\mathrm{o} = \mathrm{H}\mathbf{T}_\mathrm{o}$ and $\mathbf{T}'_\mathrm{n} = \mathrm{H}\mathbf{T}_\mathrm{n}$. This procedure avoids post-process image warping and any problems due to distortion and produces improved results, as shown in Figures 4.5c–d.

4.5 Occlusions and disocclusions

A major challenge for image-based rendering approaches are occlusions and disocclusions. In traditional image-based rendering approaches, handling occlusions requires careful reconstruction of the proxy geometry at the boundary in order to avoid artifacts due to sharp but incorrect edges. Disocclusions are similarly complex since the geometry that becomes disoccluded needs to be modeled as well, e.g., using a multi-layered representation [Shade et al., 1998; Zitnick et al., 2004].

In the gradient domain, occlusions must be handled differently, because gradients with different depths do not necessarily occlude each other; for reflective or semi-transparent surfaces, gradients on different layers just add together.

A naïve approach is to not handle occlusions at all, i.e., splat all gradients to their projected position and integrate the resulting gradient field. If the input image density is high enough, one can rely on blending between images from different viewpoints to avoid artifacts at the cost of larger use of resources. In many cases, this works surprisingly well since in particular disocclusions are filled in smoothly and consistently by this approach (see Figure 4.6, to the left of the head). However, in scenes with high amounts of occlusions, this leads to ghosting artifacts when gradients of opaque surfaces are added together. In the following, we therefore develop a more principled approach for detecting and handling occlusions and disocclusions.

4.5.1 Detection and modeling

As we cannot use depth as a sole indicator for occlusions due to reflections, we need to do an explicit search for gradients that vanish at occlusion boundaries. Given a reference view,

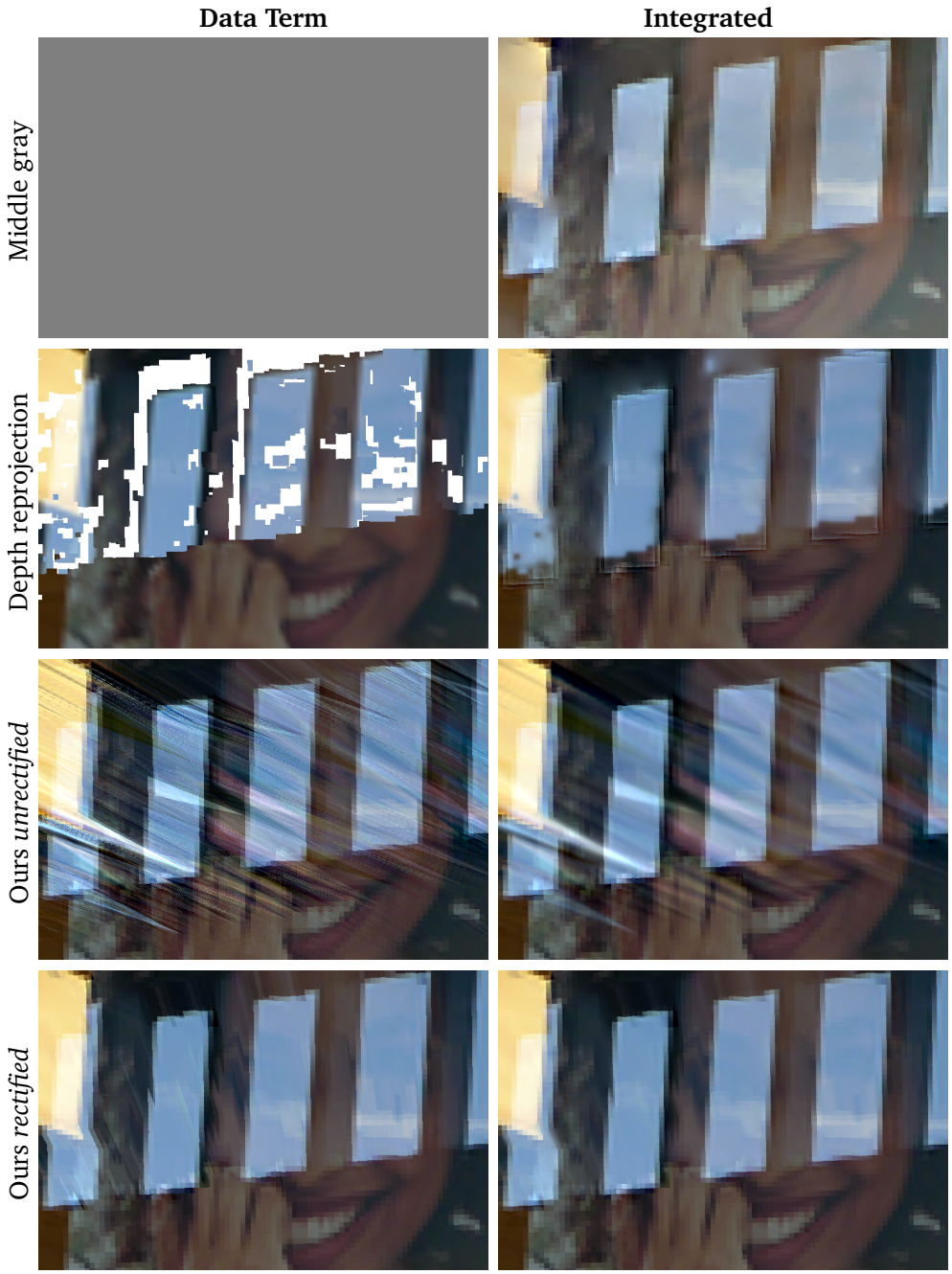


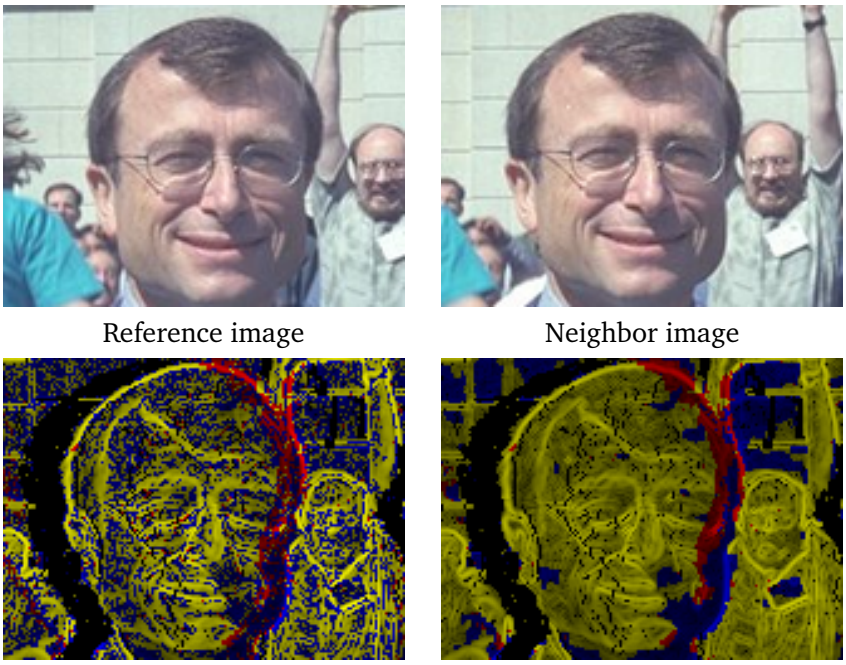
Figure 4.5: Comparing different data terms (left) for regularizing the Poisson reconstruction (right): (a) a constant middle gray data term does not always reproduce the correct colors, (b) setting the data term to a depth reprojected image produces artifacts in poorly reconstructed (typically untextured) regions, (c) our *unrectified* data term suffers from clutter, (d) our *rectified* data term yields the best results.

Input image and gradients

Novel view, occlusions *not* handled

Novel view, occlusions handled

Figure 4.6: Occlusions and disocclusions. Note how the disocclusions to the left of the neck are smoothly filled in by the Poisson integration.



Reference image | Neighbor image

Unregularized visibility | Regularized visibility

Figure 4.7: Input images and gradients classified according to their visibility in a neighboring view. Yellow: gradient visible. Blue: gradient occluded. Red: both occluded and visible gradients are projected to this location. All colors are weighted with the gradient magnitude to improve visualization.

we first detect whether its gradients are still visible in one of the neighboring views at (or near) the location predicted by the epipolar geometry. This problem is closely related to the reconstruction as described in Section 4.2.1 but requires a more robust formulation due to the fact that we need to argue locally (for a single neighboring image) and not globally for the complete set of input images.

In order to determine whether a gradient is visible in a neighboring view, we compute a similarity measure c between single-pixel gradients which is defined as the product of the angle cost $c_\angle = \mathbf{g}_1^\top \mathbf{g}_2 / \|\mathbf{g}_1\| \|\mathbf{g}_2\|$ (as introduced near Equation 4.1) and a cost c_s that measures the intensity difference between two gradients:

$$c = c_\angle^k \cdot c_s, \tag{4.7}$$

$$c_s = \min\left(\frac{\|\mathbf{g}_n\|}{\|\mathbf{g}_r\|}, \frac{\|\mathbf{g}_r\|}{\|\mathbf{g}_n\|}\right). \tag{4.8}$$

$\mathbf{g}_r$ is the gradient in the reference view at location $\mathbf{T}_\mathrm{o}\mathbf{v}$, $\mathbf{g}_n$ is the gradient in the neighboring view at the corresponding location $\mathbf{T}_\mathrm{n}\mathbf{v}$, and k determines the angular selectivity of the measure (we always use $k = 10$). The intuition behind this is that gradients should only match if they have a similar direction as well as a similar magnitude. In order to be robust to various errors (e.g., miscalibration, slightly non-planar reflectors), we do not strictly enforce the epipolar geometry. Instead of using c directly we compute $c_{max} = \max_{\mathcal{N}} c$ in a 5×5 pixel neighborhood $\mathcal{N}$ around $\mathbf{T}_\mathrm{n}\mathbf{v}$. In other words, we compute the maximal similarity score within the neighborhood $\mathcal{N}$ around the location $\mathbf{T}_\mathrm{n}\mathbf{v}$ predicted by the epipolar geometry.

To finally determine whether a given gradient is visible in each of the other views, we solve per-view a binary graph-cut with the data terms c_{max} (occluded) and $1 - c_{max}$ (visible) for the respective labels. The regularization weights between neighboring pixels are computed with a simple Potts model, assigning 0 if both labels are identical and 0.3 if they differ (see Figure 4.7).

For the rendering approach described in the next section, we not only need to determine which gradients eventually become occluded, but also the 3D position of their occluders. Since this is generally ambiguous in the reflective case, we use a simple heuristic: If a gradient $\mathbf{v}$ becomes occluded between neighboring views i and j, its occluder $\mathbf{o}$ must cross the epipolar line of $\mathbf{v}$ between i and j and it must be closer to the camera than $\mathbf{v}$. We therefore search along the epipolar line and select the first gradient with magnitude larger than 0.001 and with a smaller depth than $\mathbf{v}$ as the occluder $\mathbf{o}$.

4.5.2 Rendering

Let $\mathbf{v}$ be a gradient and $\mathbf{o}$ its occluder. If the novel camera position is not exactly on the connecting line between the input camera positions in which $\mathbf{v}$ and $\mathbf{o}$ are defined, they will

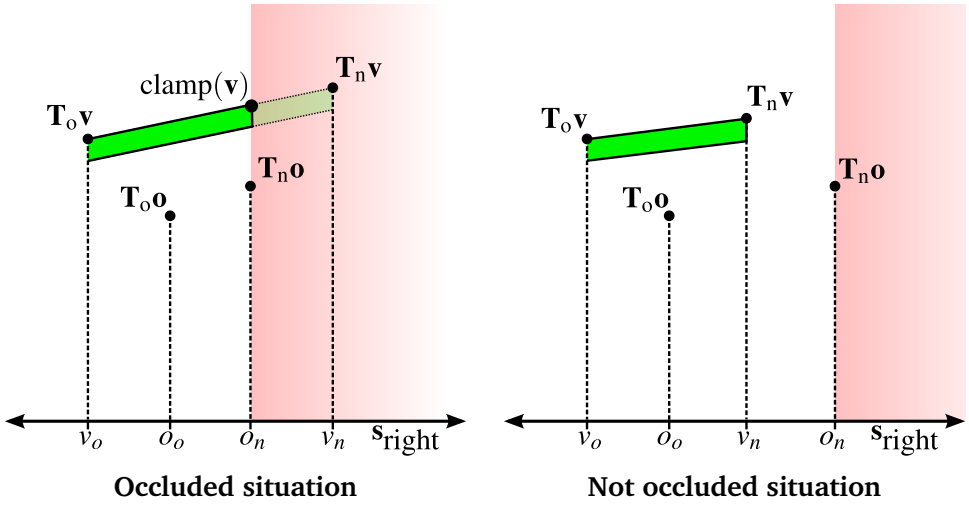


Figure 4.8: Rendering occlusions. The candidate and occluding gradient positions in the original (o) and novel (n) views are projected along the $\mathbf{s}_{\text{right}}$ vector and their values are compared. The occluded half-space is indicated by red shading. *Left:* the gradient is occluded since $v_{\text{o}} < o_{\text{o}}$ but $v_{\text{n}} > o_{\text{n}}$; the quad is clamped. *Right:* the gradient is not occluded since $v_{\text{o}} < o_{\text{o}}$ and $v_{\text{n}} < o_{\text{n}}$.

generally not intersect in screen space. Therefore, to be more robust we consider projections onto a common screen space axis $\mathbf{s}_{\text{right}}$, where $\mathbf{s}_{\text{right}}$ is the right camera axis $\mathbf{r}_{\text{right}}$ projected into screen space. The resulting original and new projections onto this axis are therefore

$$v_{\text{o}} = (\mathbf{T}_{\text{o}}\mathbf{v})^{\top}\mathbf{s}_{\text{right}}, \quad v_{\text{n}} = (\mathbf{T}_{\text{n}}\mathbf{v})^{\top}\mathbf{s}_{\text{right}}, \tag{4.9}$$

with analogous definitions for o_{o} and o_{n}.

If the order of v and o flips when comparing original and novel views, i.e.,

$$(v_{\text{o}} < o_{\text{o}}) \text{ XOR } (v_{\text{n}} < o_{\text{n}}), \tag{4.10}$$

the gradient $\mathbf{v}$ is occluded. In other words, o_{n} defines a half-space in which $\mathbf{v}$ is occluded. Figure 4.8 illustrates this situation.

When rendering the gradient terms F_x and F_y, we evaluate Equation 4.10 in the vertex shader, and simply discard occluded gradients. When rendering the approximate solution S we cannot simply discard vertices, because this fails to account for the additive sweeping effect of the partially occluded gradient (until it becomes occluded). Instead, we clamp the quad Q (see Section 4.4) against the half space defined by o_{n}, i.e., we replace $Q \leftarrow \big(\mathbf{T}_{\text{o}}\mathbf{v}_i, \mathbf{T}_{\text{o}}\mathbf{v}_j, \text{clamp}(\mathbf{v}_j), \text{clamp}(\mathbf{v}_i)\big)$, where

$$\text{clamp}(\mathbf{v}) = \mathbf{T}_{\text{o}}\mathbf{v} + (\mathbf{T}_{\text{n}}\mathbf{v} - \mathbf{T}_{\text{o}}\mathbf{v})\frac{o_{\text{n}} - v_{\text{o}}}{v_{\text{n}} - v_{\text{o}}}. \tag{4.11}$$

This clamping operation is visualized in Figure 4.8.

Render step	Average	Max	Min	St. dev.
Horz. gradients	1.056	1.242	0.738	0.174
Vert. gradients	1.132	1.313	0.814	0.170
Approx. solution	4.721	8.763	1.742	1.650
Poisson	13.343	14.033	13.105	0.358
Total	**22.616**	**27.856**	**18.761**	**2.089**

Table 4.1: Rendering times (in milliseconds) averaged across 5 different datasets showing the distribution of times according to rendering components / stages.

4.6 Results

We have developed a prototype renderer implementation in C++ using OpenGL. The Poisson integration is implemented with CUDA. We used this implementation to generate all the results presented here and in the supplementary video.

Performance: All rendering stages except the approximate solution run in constant time for a given input image size. The approximate solution stage is fill rate bound, and its performance depends on the visible length of the streaks.

In Table 4.1, we summarize timings for 5 different datasets. Each dataset uses input images of size 640×480 and is rendered at an output size of 720×480 on an NVIDIA GTX 680. For each dataset, we measured performance for ten random novel views along an arc connecting the input cameras, yielding the aggregate statistics shown in Table 4.1.

Results: We tested our approach on a variety of input sequences captured using hand-held photography under mostly horizontal camera motion. (We use primarily one-dimensional motion to make the interactive view navigation simpler, as was done by Sinha et al. [2012].) Please see our supplementary video for more results, since these show the visual artifacts much more clearly than still images.

Figure 4.9 shows view extrapolation results produced from just a single reference color image and depth map. As you can see, standard image-based rendering has great difficulty dealing with the two depths present in the wood grain table top and the reflected window. In contrast, our technique handles this reflective scene without difficulty. The fact that gradients are added allows them to interact with each other as they would on a real reflective surface. A requirement for this is of course that we predict the movement of reflected gradients correctly. This works well in most cases when the reflective object is mostly planar but it can lead to artifacts for curved surfaces.

Figure 4.10 and Figure 4.11 compare our results to Sinha et al. [2012] and show the advantage of not having to separate various layers. The separation can almost never be totally

Input image | Standard view extrapolation

Our approximate solution S | Our integrated result

Figure 4.9: View extrapolation using a single image and depth map for the SUNROOM dataset. The same proxy geometry was used for standard image-based rendering and our method.

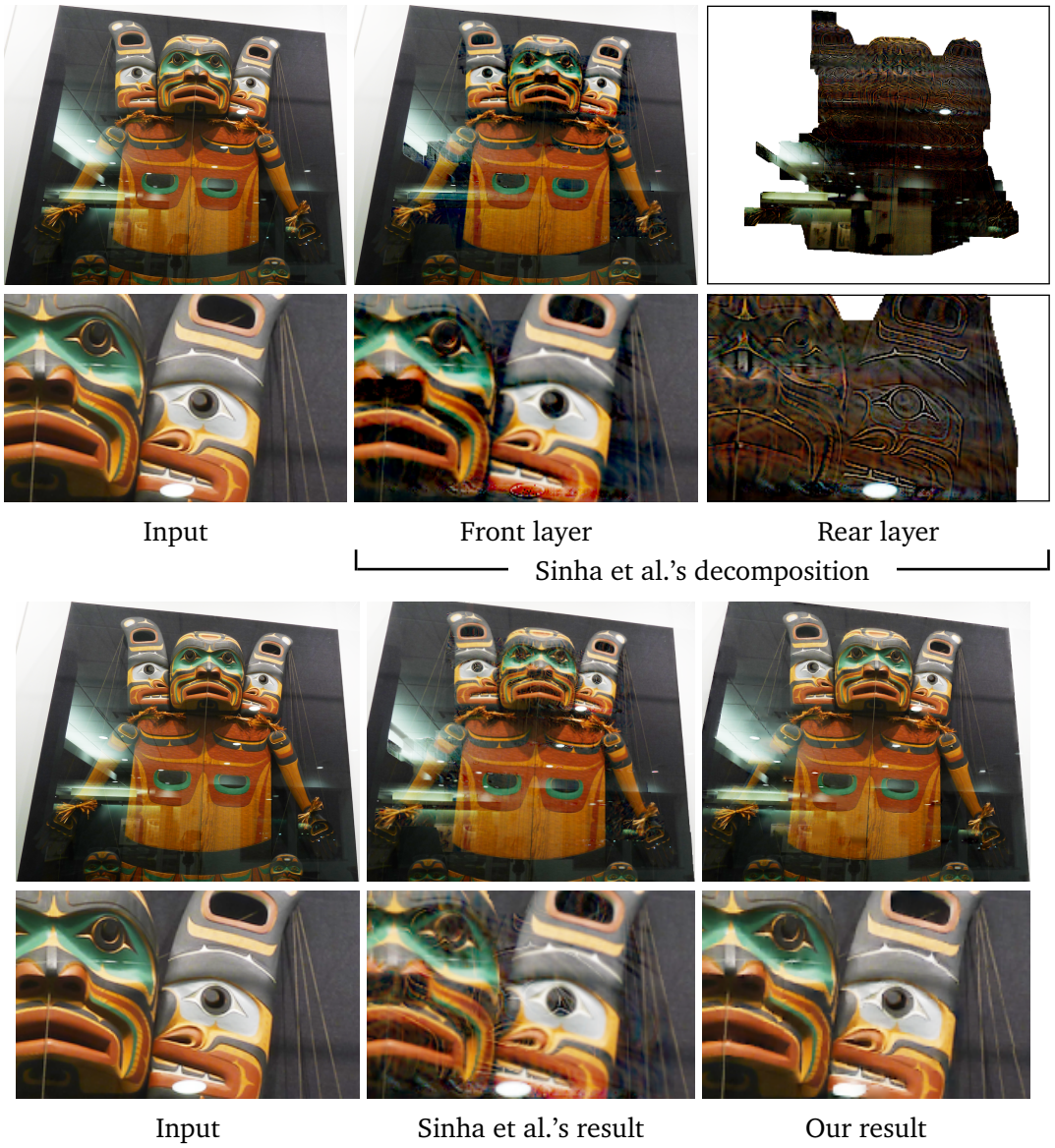


Figure 4.10: Sinha et al. [2012] explicitly decompose the input images into transmitted and reflected components. This separation is not always clean. Note the ringing artifacts in the rear layers, that are also visible in their recomposed result. Our result does not suffer from ringing artifacts.

Input image | Ground truth neighbor view

[Sinha et al., 2012] | Our result

Figure 4.11: For the MUSEUM dataset, the planar proxies of the reflection layers used by Sinha et al. cause incorrect reflections in the bottom left. Please see the video to see these artifacts more clearly.

Input image | Standard IBR

Piecewise planar solution | Our result

Figure 4.12: Input images and sample novel views (zoomed in) for the CONFERENCE dataset. The same proxy geometry was used for standard image-based rendering and our method. Please see the video for details.

correct and will therefore ultimately always lead to artifacts. In contrast our approach simply handles all gradients equally and the effect of different layers is modeled automatically.

Results for a non-reflective scene are shown in Figure 4.12[1]. Compared to regular image-based rendering or piecewise planar solutions, our technique introduces fewer artifacts when the proxy geometry contains small errors. In particular, it tends to produce higher quality for datasets with wide base lines, as long as the proxy geometry is accurate. In the supplementary video we test this by rendering a scene using only a fraction of all input images, while the proxies were computed from the full dataset.

Limitations: Compared to previous image-based rendering approaches, our new technique almost always produces improved results. However, there are still cases where it produces visible artifacts. The most common case is when the stereo algorithm associates incorrect depth values with a gradient or edge. This occurs most often in two situations: for horizontal edges, since their depth is hardest to estimate for mostly horizontal camera motion,

[1]The CONFERENCE dataset is courtesy of Dayton Taylor.

and for cluttered random textures. A challenging example for the second case is the TREE scene, which we included in the supplementary material. The occlusion detection algorithm also occasionally makes mistakes, which results in ghosted gradients and edges extending beyond the regions of a reflective surface. Finally, our rendering approach requires the use of specialized shaders and higher-end graphics hardware, which means that it may not run on all computing platforms (e.g., under-powered mobile phones).

Chapter 5

Gradient-weighted Image Warping

In this chapter we describe our new approach to regularization in image warping. Image warping is used mostly for image or video manipulation tasks such as retargeting, or stabilization. We want to apply warping to a more interactive rendering system. As warping only builds on sparse correspondences we do not need to compute an expensive dense reconstruction. This makes the technique much more suitable for mobile scenarios. However, one of the disadvantages of warping is that it does not support larger extrapolations from the input data. Extrapolations often require larger distortions in the image and previous methods such as Liu et al. [2009] struggle in these cases. To improve the performance of warping in this scenario we propose a new way of regularization. Based on the amount of gradients in the image we vary the regularization method which makes our warp more robust for larger image distortions.

5.1 Overview

As mentioned earlier image warps build on sparse correspondences. They first generate a sparse 2D mapping from an input image to a desired output image which can be done automatically or with the help of user input. The goal is then to find a warp for the whole input image based on this sparse input. For this purpose the image is split into smaller pieces, often a regular grid, and the warp is defined on this structure, see Figure 5.1. This has the advantage that the grid can be used to interpolate and regularize the sparse input. The main goals of the regularization are to introduce robustness against eventual noise or outliers in the correspondences, and to allow for and extrapolation of the warp in image regions where correspondences are completely missing. So far many algorithms simply focused on the creating better constraints from the input mapping depending on the problem at hand. However, the actual regularization has not been discussed in further detail. Almost all methods simply rely on constraints that keep the transforms of the grid cells as close as

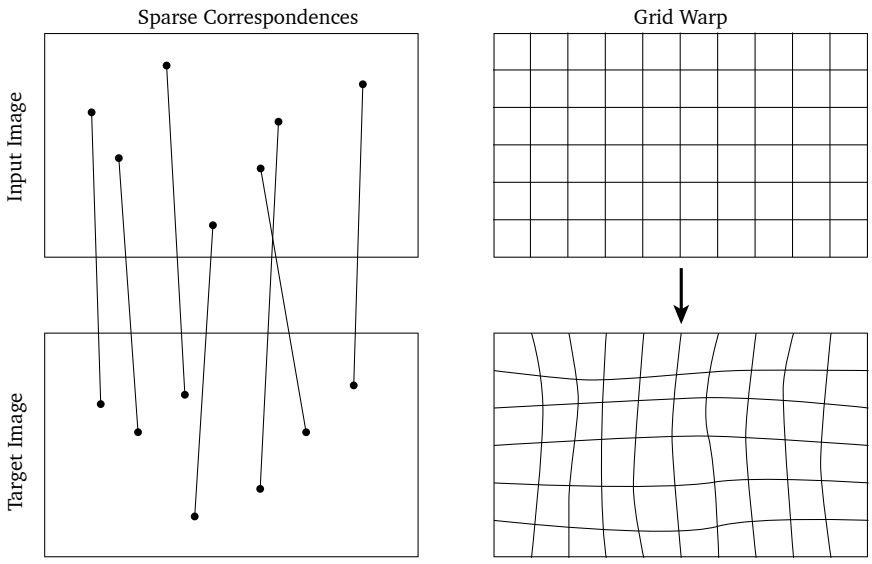


Figure 5.1: Image warp overview. For a given set of correspondences between the input and the target image (*left*) the warp is computed for the vertices of regular grid on the input image image (*right*). Depending on the correspondences the grid cells are more or less distorted.

possible to a similarity transform which means all transformations should be defined only via an isotropic scaling and a rotation. In this way all original grid cells keep their shape, and the warp does not distort the image too much. We found that in scenarios such as 3D video manipulation larger image distortions are necessary to support a larger extrapolation from the original input data. As shown by Liu et al. [2009] this quickly leads to problematic results, as the similarity constraints spread distortions over a large image area, which can lead to artifacts. We propose to relax the focus on similarity transformations and also allow affine transformations. In our experiments this leads to improved results for larger image distortions in challenging scenes. Affine constraints directly allow the input grid cells to transform into rectangles and parallelograms with varying aspect ratio. Thereby they can absorb distortions without spreading them over a large area. We use a simple heuristic to select the regions in which affine constraints can be introduced and thereby improve the visual quality of the warped results.

5.2 Related Work

Image warps are most commonly applied to problems that manipulate the image on a 2D domain. Especially image retargeting often requires more advanced warping techniques that are aware of the image content. Zhang et al. [2009] present a technique for image

resizing. Their approach is based on finding locally salient content and longer edges in the image. They construct specific constraints depending on these two types of features and finally define an energy on a regular grid that is regularized with a similarity transform. Similarly Laffont et al. [2010] present an application for interactive zooming of large images. Their goal is to always map the currently viewed subregion to a fixed screen size, which can be difficult for images with an extreme aspect ratio such as panoramas. The method is based on a multi-resolution triangle mesh that is refined for the specific area of interest to create higher quality results. To regularize the warp all distortions are also penalized using a uniform scaling analog to a similarity transform. Another retargeting approach was presented by He et al. [2013]. Here the image content is not to be scaled but to be rotated while the original image size is preserved. The method detects lines in the image and builds constraints that preserve these lines as much as possible while following a given rotation. As in Zhang et al. [2009] the warp is based on a regular grid and the shape distortion is regularized with similarity constraints.

Other approaches also perform more complex manipulations. Carroll et al. [2010] present a warp-based technique that can change the perspective in an image. They allow the user to manipulate certain lines and vanishing points in an image and construct mapping constraints based on this input. Their warp is based on a regular grid and again uses a similarity measure on the grid cells for regularization. Liu et al. [2009] use warps for 3D video stabilization. Starting with a sparse 3D reconstruction from SfM they smooth the camera path and construct feature correspondences from the input images to the new camera positions on the smoothed path. These correspondences can be used to warp the input images and create a stabilized version of the input video. This method still uses the same similarity constraints for regularization, however, compared to the previously mentioned methods this can quickly generate various artifacts. To create a realistic impression the image content needs to move according to a real 3D transformation which means that features that are far away from the camera move much slower than features close to the camera. If the smoothed camera position is far away from the original input this can quickly generate opposing constraints between image foreground and background which leads to large distortions, see Liu et al. [2009] for details. In general image-based rendering systems with denser geometry estimates (Chapter 4) silhouettes or superpixels can be used to circumvent these issues [Chaurasia et al., 2011; Chaurasia et al., 2013]. Unfortunately these approaches require additional precomputations which contradicts our approach of using only sparse 3D input that can be computed quickly. We show that we can generate better warping results in these scenarios where larger distortions are required. In contrast to all methods described above we do not rely solely on similarity constraints for regularization, but we also relax them to affine if necessary. Affine constraints were first used for warping by Chen and Gotsman [2016]. They introduce a generalized formulation that explores the connection between similarity and affine transformations. In our approach we show how their work can be used to

smoothly interpolate between the two different forms of constraints which leads to improved results for complex image warps.

5.3 Algorithm

5.3.1 Warp

The warp of the input image is defined over a regular grid of vertices $\mathbf{v}'$ and interpolated linearly for each pixel. In practice this just means creating a textured triangle mesh from the input image and rendering it with new vertex positions for the desired camera. We can formulate the computation of the new vertex positions $\mathbf{v}$ as an optimization problem consisting of a data term and a regularization term

$$\underset{\mathbf{v}}{\operatorname{argmin}} E_{\text{data}}(\mathbf{v},\mathbf{x}',\mathbf{x}) + E_{\text{reg}}(\mathbf{v}) \tag{5.1}$$

The data term is straight forward, as we want the warped grid points to approximate the sparse input mapping $\mathbf{x}' \rightarrow \mathbf{x}$ as closely as possible. Similar to Liu et al. [2009] we use bilinear weights w to connect a sparse point to its surrounding 4 vertices $\mathbf{x}'_i = w_i^T \mathbf{v}'_i$ and the data term becomes

$$E_{\text{data}} = \sum_i ||w_i^T \mathbf{v}_i - \mathbf{x}_i||^2 \tag{5.2}$$

For many scenes the data term does not cover all parts of the input image and potentially contains outliers. To mitigate these issues a sensible regularization term is necessary. Traditionally this has been done using similarity constraints. Liu et al. [2009] define this on triangles by splitting each cell of the input grid into two triangles, and requiring each triangle to undergo a similarity transform. The same constraint can also be defined on the grid cells directly as shown by He et al. [2013]. This works well for warps that do not need to introduce a lot of distortion. But for more extreme warps, simple similarity constraints can introduce strong artifacts. This is especially visible in regions without explicit data constraints. We propose to relax the constraint on similarity and only impose affine constraints in regions without image gradients.

The method of Chen and Gotsman [2016] formulates a new, more abstract constraint that can also model affine transformations. In their work they only show how affine constraints can improve small perspective image manipulations, but we found that explicitly transitioning between similarity and affine constraints can improve more extreme warps as well. We briefly summarize the main aspects of their method and refer to their publication for a detailed derivation. First, the coordinates of the vertices $\mathbf{v}$ are interpreted as complex numbers.

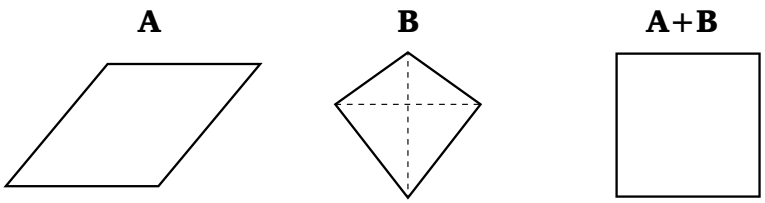


Figure 5.2: Effect of the matrix constraints **A** and **B**. **A** is a simple affine constraint that restricts the transformation to a parallelogram. **B** is the constraint for a crossgram, a quad with orthogonal diagonals of equal length. If both constraints are combined they result in a square.

A single input quad is therefore represented as a complex 4 vector $\mathbf{q}$. We can now define a quadratic energy that measures how much this quad is distorted compared to an ideal square

$$E(\mathbf{q}) = \mathbf{q}^* \mathbf{S}_{a,b} \mathbf{q}. \tag{5.3}$$

The core idea is that $S_{a,b}$ can be constructed as a linear combination of two matrices $a\mathbf{A} + b\mathbf{B}$ with

$$\mathbf{A} = \begin{pmatrix} 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \\ 1 & -1 & 1 & -1 \\ -1 & 1 & -1 & 1 \end{pmatrix} \quad \mathbf{B} = \begin{pmatrix} 1 & i & -1 & -i \\ -i & 1 & i & -1 \\ -1 & -i & 1 & i \\ i & -1 & -i & 1 \end{pmatrix} \tag{5.4}$$

Both of these matrices represent separate constraints. **A** constraints the quad to be a parallelogram so it stands for a purely affine constraint. **B** models a constraint that forces the quad to be a so-called crossgram, which is a quad with orthogonal diagonals that have equal length. The sum of both then represents a square which is equal to a similarity constraint, see Figure 5.2.

Chen and Gotsman [2016] showed that these affine constraints can be helpful for perspective image manipulation. We use the fact that this formulation can also smoothly interpolate between affine and similarity constraints. Since **A** purely represents an affine constraint we can add a varying amount of **B** to transition more towards a similarity constraint. Our energy for the regularization constraints is therefore

$$E_{\text{reg}}(\mathbf{v}) = \mathbf{v}^* \mathbf{S}_\alpha \mathbf{v} \quad \text{with} \quad \mathbf{S}_\alpha = \mathbf{A} + \alpha \mathbf{B} \tag{5.5}$$

We propose to choose the variable α based on image content. Since we want to support larger extrapolations from the original image we usually need to introduce many distortions on the warp grid. To hide these distortions as much as possible we aim to add them only to regions without strong texture. We use a simple heuristic and compute the sum of the image gradient magnitudes in each grid cell. After normalizing these sums over all cells to

Figure 5.3: Example distribution of our weight α. *Left*: Input image. *Right*: Weights computed for each grid cell (white is higher).

the interval $[0.001, 1]$ they can be directly used as α values. Note that α is never exactly 0 as we found that leaving a small lower bound leads to slightly better results. Figure 5.3 shows an example of the computed distribution of α.

While Liu et al. [2009] followed a similar approach, they only changed the weight on their similarity constraints. This often leads to huge distortions at the border of the image where strong data constraints are missing. Similarity constraints cannot really absorb distortions because every quad still always tries to stay a square. Any deviation from that generates an error, and even if the weight on this is lowered the error will affect neighboring quads. In comparison an affine constraint has more degrees of freedom and can therefore model a much greater set of distortions without introducing any form of error.

5.3.2 Rendering Setup

All of our inputs are short videos, captured with a handheld smartphone. This is the easiest and most straight forward way of capturing data for many casual users. For our application we built an interactive viewing experience based on a regular sparse scene reconstruction. We first run a general SfM method that generates a set of views V_i with associated images and camera projection matrices, a set of sparse 3D points p_i, and a mapping that provides a subset of visible points for each of the input views. To create a rendering from a new camera pose we select the closest input view and project all visible points from nearby cameras into both the view and the novel camera. This creates a sparse 2D to 2D mapping from points in the input image $\mathbf{x}'$ to points in the target image $\mathbf{x}$, representing the 3D motion of all selected sparse points, see Figure 5.4. We now have to find a warp that transforms the input image according to these constraints while introducing as few artifacts as possible. This is similar to the camera stabilization algorithm presented by Liu et al. [2009], but we want to enable larger deviations from the input cameras to allow for more flexibility during rendering. Compared to our approach from Chapter 4, this method generates an image-based rendering without an explicit dense reconstruction step. Note that when combined

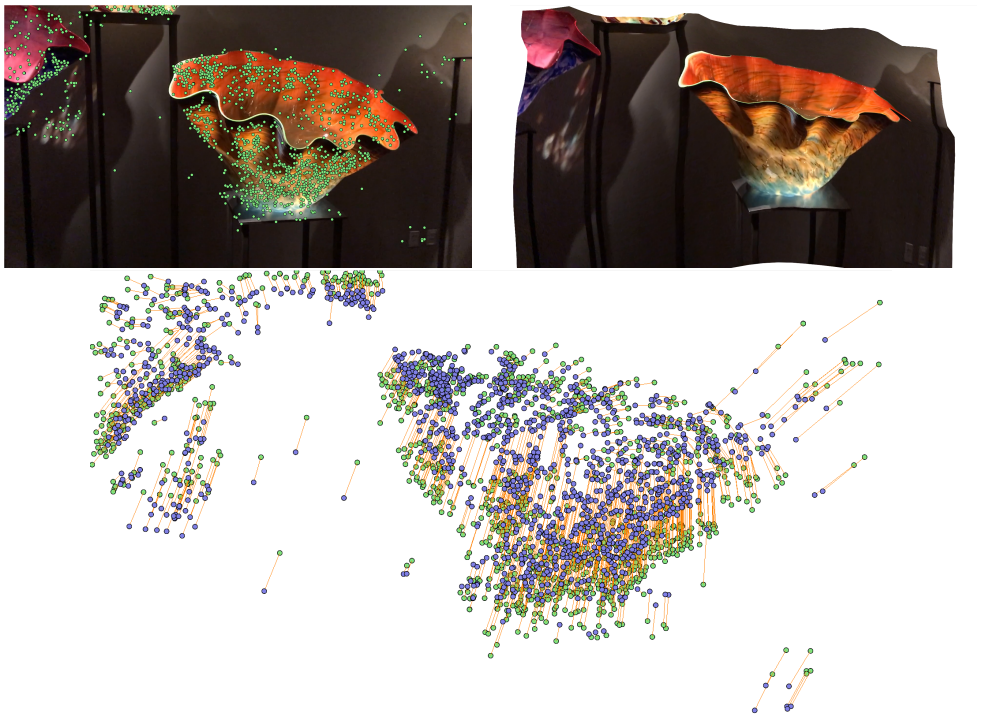


Figure 5.4: *Top left*: Input image with original sparse features ●. *Bottom*: Projecting the features into a new view pose ● creates a 2D to 2D mapping. *Top right*: Output image warped according to the 2D constraints.

with a real-time sparse reconstruction method (e.g. the ones available in AR toolkits on modern mobile platforms [1] [2]) this approach could also create results instantly after capturing.

Navigation To demonstrate our warp we created a basic navigation mechanism for input videos captured on a continuous path, i.e. the camera moves into roughly the same direction for the whole sequence. Starting from the sparse reconstruction we create a smoothed version of the camera path by fitting a centripetal Catmull–Rom spline. We also create a smooth path for the camera look-at point. For this path we formulate an optimization problem that minimizes the path length and the distance of the projection to the optical center of the original cameras. Figure 5.5 shows an example scene with the fitted camera and look-at paths. Finally the navigation is modeled with two degrees of freedom. One dimension is associated with the original camera path and the other dimension is associated with the up-vector of the cameras. The distance to the original camera path is also clamped at a certain value because the extrapolation with our warp is still limited. For the supplementary video we generate automated results that explore the available camera positions by moving the virtual viewpoint along a sine curve.

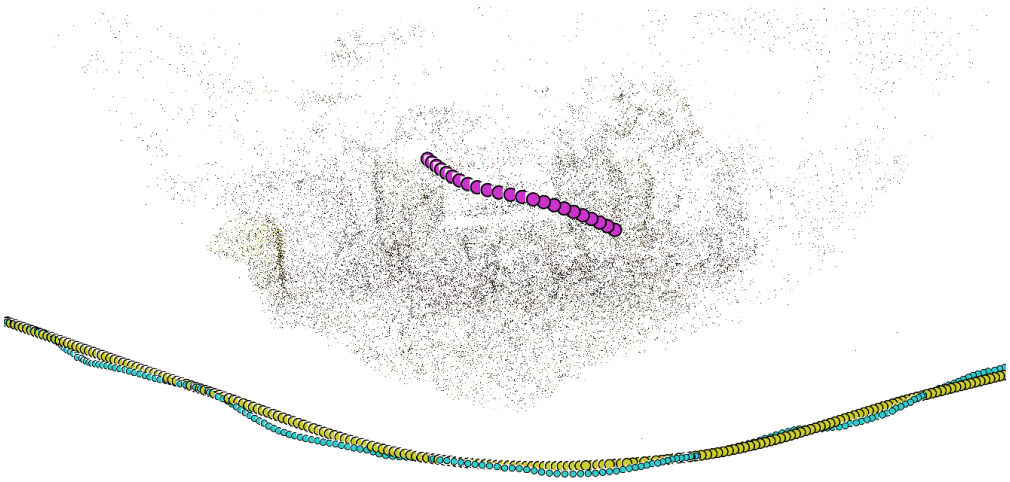


Figure 5.5: Example scene structure with sparse reconstruction from SfM and camera positions. The original cameras ● are interpolated and smoothed with a spline ●. The look-at positions for the cameras are also smoothed with a separate spline ●.

[1] https://developer.apple.com/arkit/
[2] https://developers.google.com/ar/discover/

5.4 Results

We have implemented our prototype application in C++ using basic OpenGL rendering. Constructing and solving the linear energy minimization is done single threaded on the CPU. For an average scene and a 30×30 grid computing the warp takes 30-40ms and rendering the final image is done with basic texture mapping in less than 1ms. While this already runs in real time, parallelizing the linear solver or moving the optimization completely onto the GPU can decrease the time spent for solving the system significantly. Higher frame rates or an application on mobile devices are therefore very feasible.

To evaluate our method we compare our warping technique against basic similarity constraints that are used in previous work. For a fair comparison we also weight the similarity constraints according to the image content similar to Liu et al. [2009]. As our application is focused on interactive 3D rendering we do not directly compare against image retargeting approaches. For all of our scenes we show screen captures from our prototype renderer and we also show automatically generated interactions in the supplementary video. All of our input data is extracted from short videos that are captured handheld with a smartphone. The videos are captured on a horizontal arc and we then extrapolate the vertical movement. To compute the camera path we first extract all frames from the video and run a regular SfM pipeline. For all results please also see our supplementary video for visualizations of the real-time rendering.

In our first scene *Glass* we demonstrate an extrapolation from a single frame. This scene has some generally difficult properties: The reflective surface of the glass generates an unusually high amount of outlier matches in the sparse reconstruction and the featureless background also results in larger areas without strong data constraints (see also Figure 5.4). As shown in Figure 5.6 our regularization constraints provide a clear advantage compared to standard similarity constraints. Especially in regions without strong data constraints our constraints reduce the amount of distortion. The warp grid shows that the similarity constraints force each cell to be a square. While cells in areas with low texture can deviate from that due to the weights on the constraints, they are still forced to be a square as much as possible. The results is that the distortion is distributed over many cells resulting in large folds (also shown by Liu et al. [2009]). In contrast our constraints are reduced to affine for areas without texture. This allows certain cells to absorb larger amounts of distortion as they can be deformed more without adding additional costs. We can see that some cells are clearly distorted more towards a parallelogram which allows other cells to better retain there original shape.

The second scene *Statue* demonstrates the same effect on a larger dataset. Figure 5.7 shows the extrapolation for a single input image. The distortions created by the similarity constraints are again visible in regions without strong data constraints. For the center of the image both approaches are comparable as many data constraints are generated from the

sparse reconstruction. In this case the regularization does not affect the warp as it its already clearly defined by the mapping from the sparse points.

A third scene *Sculpture* shows another example where our constraints performs better at extrapolation. We show the results in Figure 5.8. This scene is also more challenging as the sparse reconstruction does not work well for the reflective background of the scene, which leads to areas without reconstructed features and also more outliers.

Here and in the previous scene we can also see the limitations of our method. They are especially visible in the extrapolations of the whole sequence that we show in the supplementary video. A great disadvantage of the warping approach is that the grid stays connected and can only be deformed. This cannot model the difference between foreground and background properly which often leads to large distortions. Our regularization can handle these distortions better than previous approaches but it cannot remove them. For this to be possible we would need an explicit, pixel-wise separation between foreground and background layers. This is in general a difficult problem which cannot be solved properly in real-time. As we want our method to only rely on a sparse scene reconstruction we are not able to handle this problem efficiently.

Figure 5.6: Extrapolation of a single image for the *Glass* scene. Compared to the input image the camera is moved downwards, away from the input data. (Best viewed on a screen.) *Top*: Input image. *Middle*: Extrapolation using similarity constraints for regularization; warp grid visualization on the right. *Bottom*: Extrapolation using our mixed constraints; warp grid visualization on the right.

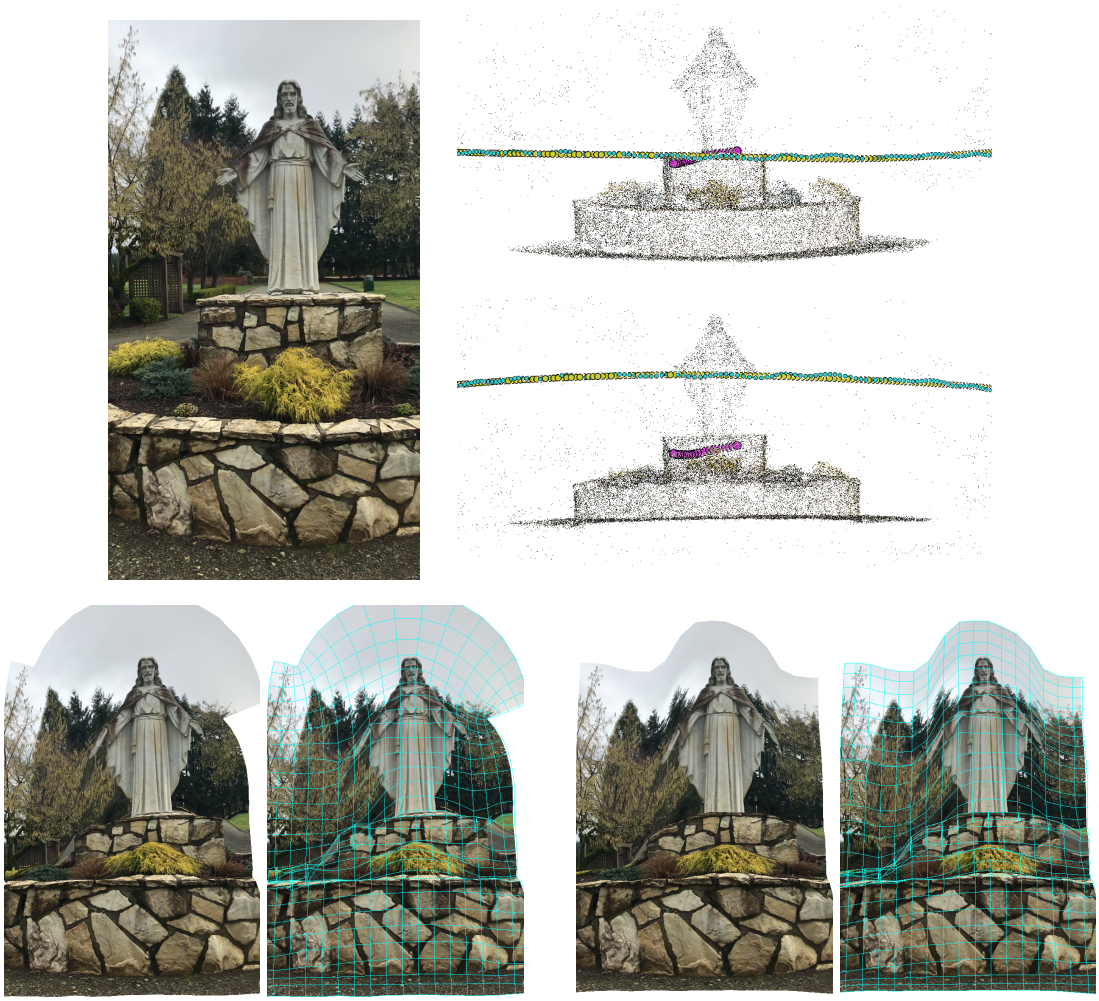


Figure 5.7: Extrapolation of a single image for the *Statue* scene. (Best viewed on a screen.) *Top*: Input image and sparse scene reconstruction from original camera position and extrapolated camera position. *Bottom left*: Extrapolation using similarity constraints for regularization; warp grid visualization on the right. *Bottom right*: Extrapolation using our mixed constraints; warp grid visualization on the right.

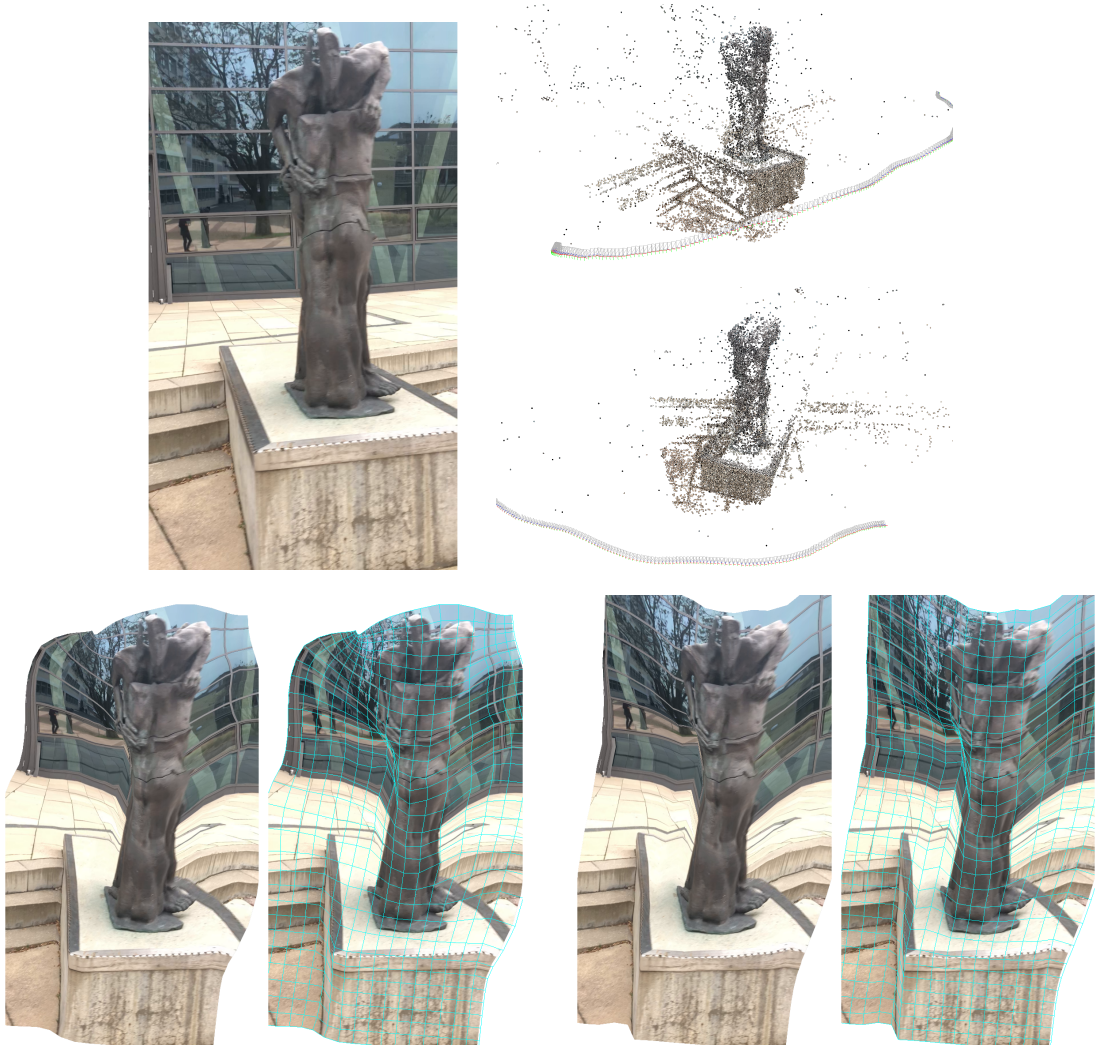


Figure 5.8: Extrapolation of an image for the *Sculpture* scene. *Top*: Input image and sparse scene reconstruction showing only points and original cameras from two perspectives. The complex background of the scene leads to few reconstructed features behind the sculpture. *Bottom left*: Extrapolation using similarity constraints for regularization; warp grid visualization on the right. The missing constraints and some in the background lead to larger distortions in the warp. *Bottom right*: Extrapolation using our mixed constraints; warp grid visualization on the right.

Chapter 6

Conclusion

In this thesis we have presented multiple new techniques that push the limits of image-based reconstruction and rendering. The overall idea is that using image gradients implicitly avoids problems that previous pixel based algorithms had. They provide a more general way of handling surface properties during reconstruction and in rendering they lead to much fewer artifacts for erroneous or incomplete input data. Our specific contributions have been achieved with 3 distinct methods:

- Our novel multi-view stereo algorithm efficiently combines a stereo error term with a shading-based error term in a single, combined optimization. Building on the Retinex assumption, we are able to completely remove the albedo from the shading-based error, which has not been done before. The energy formulation relies solely on pixel-wise data terms and does not need to employ explicit regularization.

- Our image-based rendering algorithm associates depths with image gradients and reconstructs both a novel gradient image and a lower-fidelity guide image from the gradient motions. Compared to previous systems, which associate depths and colors with each pixel, this can greatly improve the quality of the rendering, especially for reflective surfaces. It also mitigates depth errors for areas with low gradients as they do not contribute much to the final solution.

- Our new image warping regularization provides a more stable way for distributing distortions. By allowing a transition between similarity and affine energies we can create more stable warps on image boundaries and in regions without strong data constraints.

Overall our gradient domain formulations can make several improvements over previous methods. Especially regarding 3D reconstruction and full image-based rendering we showed that our focus on gradients leads to more elegant solutions that handle complex issues automatically. In our multi-view stereo approach we arrive at a formulation for a shading based

energy that is solely based on geometry. Previous methods always needed to also model the surface albedo explicitly [Zollhöfer et al., 2015]. These models are often based on heuristics that can fail easily. As gradients can be separated more clearly our formulation is more robust. It allows us to focus two different energies towards their respective strengths. A similar result can be seen for our image-based rendering approach. Here our formulation automatically handles reflective surfaces without modeling them explicitly. As gradients are mostly caused by a specific change in surface color or geometry they automatically separate reflective and reflected layers; again previous methods had to perform this separation in an explicit error-prone step [Sinha et al., 2012]. This also means that most of the important changes in the scene are captured by the gradients. As we finally integrate the rendered gradients our method also focuses on the important changes automatically. Therefore reconstruction errors in areas without gradients (where they usually happen more often) do not affect our final output.

In summary we show that gradients can offer a distinct advantage in multiple areas of computer vision and graphics which suggests that they might be useful for more applications. Especially in rendering it is always useful to focus the majority of the computation on areas where the most change happens. We also know that stereo works best for stronger gradients and using shading based techniques in other areas should be adopted in other approaches as well. To some extent this is already happening. So-called event cameras specifically record only changes in the image instead of pixel values [Brandli et al., 2014] which has been shown to provide advantages for vision applications [Kim et al., 2016]. In rendering, gradients have also been adopted for global illumination [Kettunen et al., 2015].

From a content creation perspective our methods offer a variety of use cases that cover a wide spectrum of possible applications. For example in AR it is more common to render specific objects in a new context. Here our MVS algorithm can be used to automatically create such objects in very high quality. The reconstructed triangle meshes can be textured [Waechter et al., 2014] and used for traditional rasterization techniques. For efficient rendering it would also be possible to only create a low resolution mesh of the reconstruction and explicitly apply the high-quality normal maps our method creates to generate detailed images. Of course the reconstructions can also be used for other more direct applications in cultural heritage, biology, or architecture. Our IBR technique can be applied to create a realistic VR environment. The outputs of the method offer the most complete representation of an entire scene and it is also the most robust approach for very complex surface materials and scene structures. With further optimizations it is also realistic to achieve very high frame rates that are usually required for a smooth experience. The warping algorithm is mostly suited for mobile applications where it is not possible to spend time on precomputations and where the compute power is more limited. With recent advancements in real time sparse reconstruction both iOS and Android now offer AR toolkits that directly provide the input data for the warp. Therefore it should be straight forward to use our technique in a real time

mobile application without specific adjustments.

Apart from the core research we also put a focus on making some of our methods available to the public. As mentioned earlier content creation is a large focus for many applications and there is a strong demand for usable and open software in this domain. For our work we especially focused on providing an open and free environment for camera calibration with SfM and detailed 3D reconstruction with MVS as they are also required for further image-based rendering methods. We released our software under a permissive BSD licence [1] [2] and we do not require external libraries. This combination is especially useful for commercial applications as it does not require them to adopt a specific licence. We have already received a lot of feedback from different companies and other research groups that actively use our code. Their applications range from detailed forensic facial reconstruction [3] to large scale aerial surveys [4].

Future Work

General directions for future work can be derived directly from the advantages that our methods have compared to traditional approaches. For example it is generally easy for humans to distinguish between shading effects and surface albedo, yet this has been a very hard problem in computer vision. Our general insight is that solving this can be done more naturally by having multiple gradient based energies that complement each other and model the complete image formation process instead of just a single part. Starting from this point the idea can be applied in many other similar reconstruction algorithms. This improves not only the accuracy of the reconstruction but also the robustness for surface areas where having only one type of energy is not sufficient. We can also derive direct implications from our image-based rendering techniques. A single pixel value can be affected by many different lighting effects especially for complex surface materials. Similar to how reconstruction approaches try to separate shading and albedo some rendering techniques also try to separate the scene into multiple layers or other more complicated constructs. While this can help initially it will always lead to artifacts when this separation is wrong or cannot be determined unambiguously. Our solution shows that this process is not really necessary if we focus on image gradients. Gradients are naturally sparse and directly reveal changes in surface geometry or reflectance; rendering them automatically handles complex effects which can be explored in further applications.

Immediate steps can also be taken to improve upon and overcome the current limitations of

[1] https://github.com/simonfuhrmann/mve
[2] https://github.com/flanggut/smvs
[3] https://github.com/flanggut/smvs/issues/24
[4] https://skycatch.com

our algorithms.

Multi-view Stereo Our current algorithm cannot handle complex lighting scenarios so incorporating more complex lighting models would be a direct improvement. It would be especially useful to combine information from multiple images in order to create a more global lighting model that can simulate direct shadows in some form. Considering that the geometry of the object itself can create shadows it will become hard to efficiently include this model in the optimization. At this point it can also help to define a global surface model for the scene instead of focusing on a single surface for each image. A global 3D surface can then be discretized with triangles instead of patches and our optimization energy can be reused directly.

Image-based rendering Our IBR approach is currently focused on view interpolation but it could be extended towards a more general algorithm like the unstructured Lumigraph. While it is directly possible to reproject gradients instead of pixels it can be more complicated to generate an approximate solution as our method of moving the gradients is more targeted towards a single image. In terms of scene properties our approach is already good at handling textureless areas and some reflections but it cannot yet render scenes with reflective surfaces that are not planar. In this case the movement of the reflected gradients depends on the geometry of the reflective surface. While this is a very complex interaction, modeling it during reconstruction and rendering could be very helpful for more complex scenes.

Image warping Lastly, our image warping is also still limited to smaller amounts of distortion. The general structure of the regular grid on the image simply cannot handle large occlusions or disocclusions. A direct way to solve this would be to combine and blend the data from multiple images but that requires more complex warping constraints to ensure a consistent reprojection from all images. Especially if we want to keep the real time aspect from reconstruction to rendering the sparse input data limits the amount of constraints that can be applied.

Bibliography

Adelson, Edward H. and James R. Bergen (1991). “The Plenoptic Function and the Elements of Early Vision.” In: *Computational Models of Visual Processing*. MIT Press.

Agarwal, Sameer, Noah Snavely, Steven M. Seitz, and Richard Szeliski (2010). “Bundle Adjustment in the Large.” In: *Proceedings of the 11th European Conference on Computer Vision: Part II*. ECCV’10. Heraklion, Crete, Greece: Springer-Verlag.

Aroudj, Samir, Patrick Seemann, Fabian Langguth, Stefan Guthe, and Michael Goesele (2017). “Visibility-Consistent Thin Surface Reconstruction Using Multi-Scale Kernels.” In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*. Vol. 36. 6. ACM.

Barnard, Stephen T. and Martin A. Fischler (1982). “Computational Stereo.” In: *ACM Comput. Surv.* 14.4.

Bay, Herbert, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool (2008). “Speeded-Up Robust Features (SURF).” In: *Comput. Vis. Image Underst.* 110.3.

Beeler, Thabo, Bernd Bickel, Paul Beardsley, Bob Sumner, and Markus Gross (2010). “High-quality Single-shot Capture of Facial Geometry.” In: *ACM Trans. Graph.* 29.4.

Beery, Efrat and Arie Yeredor (2008). “Blind Separation of Superimposed Shifted Images Using Parameterized Joint Diagonalization.” In: *IEEE Transactions on Image Processing* 17.3.

Bergen, James R., Peter J. Burt, Rajesh Hingorani, and Shmuel Peleg (1992). “A Three-Frame Algorithm for Estimating Two-Component Image Motion.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 14.9.

Bhat, Dinkar N. and Shree K. Nayar (1998). “Stereo and Specular Reflection.” In: *International Journal of Computer Vision* 26.2.

Black, Michael J. and Anand Rangarajan (1996). “On the Unification of Line Processes, Outlier Rejection, and Robust Statistics with Applications in Early Vision.” In: *Int. J. Comput. Vision* 19.1.

Bleyer, Michael, Christoph Rhemann, and Carsten Rother (2011). “PatchMatch Stereo - Stereo Matching with Slanted Support Windows.” In: *Proceedings of the British Machine Vision Conference*.

Blinn, James F. and Martin E. Newell (1976). “Texture and Reflection in Computer Generated Images.” In: *Commun. ACM* 19.10.

Boykov, Yuri, Olga Veksler, and Ramin Zabih (2001). “Fast Approximate Energy Minimization via Graph Cuts.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 23.11.

Brandli, C., R. Berner, M. Yang, S. C. Liu, and T. Delbruck (2014). “A 240×180 130 dB 3μs Latency Global Shutter Spatiotemporal Vision Sensor.” In: *IEEE Journal of Solid-State Circuits* 49.10.

Buehler, Chris, Michael Bosse, Leonard McMillan, Steven J. Gortler, and Michael F. Cohen (2001). “Unstructured Lumigraph Rendering.” In: *Proc. SIGGRAPH 2001*.

Carceroni, Rodrigo L. and Kiriakos N. Kutulakos (2002). “Multi-View Scene Capture by Surfel Sampling: From Video Streams to Non-Rigid 3D Motion, Shape and Reflectance.” In: *International Journal of Computer Vision* 49.2/3.

Carroll, Robert, Aseem Agarwala, and Maneesh Agrawala (2010). “Image Warps for Artistic Perspective Manipulation.” In: *ACM SIGGRAPH 2010 Papers*. SIGGRAPH ’10. ACM.

Chai, Menglei, Linjie Luo, Kalyan Sunkavalli, Nathan Carr, Sunil Hadap, and Kun Zhou (2015). “High-quality Hair Modeling from a Single Portrait Photo.” In: *ACM Trans. Graph.* 34.6.

Chaurasia, Gaurav, Sylvain Duchene, Olga Sorkine-Hornung, and George Drettakis (2013). “Depth Synthesis and Local Warps for Plausible Image-based Navigation.” In: *ACM Transactions on Graphics* 32.3.

Chaurasia, Gaurav, Olga Sorkine, and George Drettakis (2011). “Silhouette-aware Warping for Image-based Rendering.” In: *Proceedings of the Twenty-second Eurographics Conference on Rendering*. EGSR ’11, pp. 1223–1232.

Chen, Qifeng and V. Koltun (2013). “A Simple Model for Intrinsic Image Decomposition with Depth Cues.” In: *Computer Vision (ICCV), 2013 IEEE International Conference on*.

Chen, Renjie and Craig Gotsman (2016). “Generalized As-Similar-As-Possible Warping with Applications in Digital Photography.” In: *Comput. Graph. Forum* 35.2.

Chen, Shenchang and Lance Williams (1993). “View Interpolation for Image Synthesis.” In: *Proc. SIGGRAPH ’93*.

Cohen, Michael F., John Wallace, and Pat Hanrahan (1993). *Radiosity and Realistic Image Synthesis*. San Diego, CA, USA: Academic Press Professional, Inc.

Collins, R. T. (1995). *A Space-Sweep Approach to True Multi-Image Matching*. Tech. rep. Amherst, MA, USA.

Criminisi, Antonio, Sing Bing Kang, Rahul Swaminathan, Richard Szeliski, and P. Anandan (2005). "Extracting Layers and Analyzing their Specular Properties Using Epipolar-Plane-Image Analysis." In: *Computer Vision and Image Understanding* 97.1.

Debevec, Paul (2008). "Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-based Graphics with Global Illumination and High Dynamic Range Photography." In: *ACM SIGGRAPH 2008 Classes*. SIGGRAPH '08. Los Angeles, California: ACM.

Debevec, Paul E. and Jitendra Malik (1997). "Recovering High Dynamic Range Radiance Maps from Photographs." In: *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '97. ACM Press/Addison-Wesley Publishing Co.

Debevec, Paul E., Camillo J. Taylor, and Jitendra Malik (1996). "Modeling and Rendering Architecture from Photographs: A hybrid geometry- and image-based approach." In: *Proc. SIGGRAPH '96*.

Dhond, U. R. and J. K. Aggarwal (1989). "Structure from stereo-a review." In: *IEEE Transactions on Systems, Man, and Cybernetics* 19.6.

Diamant, Yaron and Yoav Y. Schechner (2008). "Overcoming Visual Reverberations." In: *Proc. Computer Vision and Pattern Recognition (CVPR'08)*.

Eisemann, Martin, Bert De Decker, Marcus Magnor, Philippe Bekaert, Edilson de Aguiar, Naveed Ahmed, Christian Theobalt, and Anita Sellent (2008). "Floating Textures." In: *Computer Graphics Forum (Proc. Eurographics 2008)* 27.2.

Or-El, R., G. Rosman, A. Wetzler, R. Kimmel, and A. M. Bruckstein (2015). "RGBD-fusion: Real-time high precision depth recovery." In: *Computer Vision and Pattern Recognition (CVPR)*.

Fairchild, Mark D. (2013). *Color Appearance Models*. Third. Wiley.

Fischler, Martin A. and Robert C. Bolles (1981). "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography." In: *Commun. ACM* 24.6.

Fuhrmann, Simon and Michael Goesele (2014). "Floating Scale Surface Reconstruction." In: *Proceedings of ACM SIGGRAPH*.

Fuhrmann, Simon, Fabian Langguth, and Michael Goesele (2014). "MVE - A Multiview Reconstruction Environment." In: *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage (GCH)*. Vol. 6. 7.

Fuhrmann, Simon, Fabian Langguth, Nils Moehrle, Michael Waechter, and Michael Goesele (2015). “MVE - An image-based reconstruction environment.” In: *Computers & Graphics* 53.

Furukawa, Y., B. Curless, S. M. Seitz, and R. Szeliski (2010). “Towards Internet-scale multi-view stereo.” In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*.

Furukawa, Yasutaka and Carlos Hernández (2015). “Multi-View Stereo: A Tutorial.” In: *Foundations and Trends in Computer Graphics and Vision* 9.

Furukawa, Yasutaka and Jean Ponce (2009). “Carved Visual Hulls for Image-Based Modeling.” In: *Int. J. Comput. Vision* 81.1.

Furukawa, Yasutaka and Jean Ponce (2010). “Accurate, Dense, and Robust Multi-View Stereopsis.” In: *Transactions on Pattern Analysis and Machine Intelligence (PAMI)* 32.8.

Galliani, S., K. Lasinger, and K. Schindler (2015). “Massively Parallel Multiview Stereopsis by Surface Normal Diffusion.” In: *2015 IEEE International Conference on Computer Vision (ICCV)*.

Glassner, Andrew S. (1994). *Principles of Digital Image Synthesis*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Goesele, M., N. Snavely, B. Curless, H. Hoppe, and S.M. Seitz (2007). “Multi-View Stereo for Community Photo Collections.” In: *International Conference on Computer Vision (ICCV)*.

Goesele, Michael, Jens Ackermann, Simon Fuhrmann, Carsten Haubold, Ronny Klowsky, Drew Steedly, and Richard Szeliski (2010). “Ambient point clouds for view interpolation.” In: *ACM Transactions on Graphics (Proc. SIGGRAPH 2010)* 29.4.

Goesele, Michael, Brian Curless, and Steven M. Seitz (2006). “Multi-View Stereo Revisited.” In: *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*. CVPR ’06. IEEE Computer Society.

Goesele, Michael, Xavier Granier, Wolfgang Heidrich, and Hans-Peter Seidel (2003). “Accurate Light Source Acquisition and Rendering.” In: *ACM SIGGRAPH 2003 Papers*. SIGGRAPH ’03. San Diego, California: ACM.

Gortler, Steven J., Radek Grzeszczuk, Richard Szeliski, and Michael F. Cohen (1996). “The Lumigraph.” In: *Proc. SIGGRAPH ’96*.

Greene, Ned (1986). “Environment Mapping and Other Applications of World Projections.” In: *IEEE Comput. Graph. Appl.* 6.11.

Grosse, Roger, Micah K Johnson, Edward H Adelson, and William T Freeman (2009). "Ground truth dataset and baseline evaluations for intrinsic image algorithms." In: *Computer Vision*. IEEE.

Han, Yudeog, Joon-Young Lee, and In So Kweon (2013). "High Quality Shape from a Single RGB-D Image under Uncalibrated Natural Illumination." In: *Computer Vision (ICCV), 2013 IEEE International Conference on*.

Hartley, R. I. and A. Zisserman (2004). *Multiple View Geometry in Computer Vision*. Second. Cambridge University Press.

Hartley, Richard I. (1997). "In Defense of the Eight-Point Algorithm." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 19.6.

He, K., H. Chang, and J. Sun (2013). "Content-Aware Rotation." In: *2013 IEEE International Conference on Computer Vision*.

Heise, P., B. Jensen, S. Klose, and A. Knoll (2015). "Variational PatchMatch MultiView Reconstruction and Refinement." In: *2015 IEEE International Conference on Computer Vision (ICCV)*.

Hernandez Esteban, Carlos, George Vogiatzis, and Roberto Cipolla (2008). "Multiview Photometric Stereo." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 30.3.

Hiep, V. H., R. Keriven, P. Labatut, and J. P. Pons (2009). "Towards high-resolution large-scale multi-view stereo." In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*.

Hirschmüller, Heiko (2005). "Accurate and Efficient Stereo Processing by Semi-Global Matching and Mutual Information." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. CVPR '05. IEEE Computer Society.

Hirschmüller, Heiko (2008). "Stereo Processing by Semiglobal Matching and Mutual Information." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.2.

Hirschmuller, Heiko and Daniel Scharstein (2009). "Evaluation of Stereo Matching Costs on Images with Radiometric Differences." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 31.9.

Horn, B.K.P. (1974). "Determining Lightness from an Image." In: *Computer Graphics and Image Processing* 3.1.

Hosni, Asmaa, Michael Bleyer, Christoph Rhemann, Margrit Gelautz, and Carsten Rother (2011). "REal-time Local Stereo Matching Using Guided Image Filtering." In: *Proceedings of the 2011 IEEE International Conference on Multimedia and Expo*. ICME '11. IEEE Computer Society.

Irani, Michal, Benny Rousso, and Shmuel Peleg (1994). "Computing Occluding and Transparent Motions." In: *International Journal of Computer Vision* 12.1.

Ju, Shanon X., Michael J. Black, and Allan D. Jepson (1996). “Skin and Bones: Multi-layer, Locally Affine, Optical Flow and Regularization with Transparency.” In: *Proc. Computer Vision and Pattern Recognition (CVPR ’96)*.

Kanade, T. and M. Okutomi (1994). “A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 16.9.

Kanade, Takeo, Atsushi Yoshida, Kazuo Oda, Hiroshi Kano, and Masaya Tanaka (1996). “A Stereo Machine for Video-Rate Dense Depth Mapping and Its New Applications.” In: *Proceedings of the 1996 Conference on Computer Vision and Pattern Recognition (CVPR ’96)*. CVPR ’96. IEEE Computer Society.

Kang, Sing Bing, Yin Li, Xin Tong, and Heung-Yeung Shum (2006). “Image-based Rendering.” In: *Foundations and Trends in Computer Graphics and Vision* 2.

Kang, Sing Bing, R. Szeliski, and P. Anandan (2000). “The geometry-image representation tradeoff for rendering.” In: *Proceedings 2000 International Conference on Image Processing*. Vol. 2.

Kannala, Juho and Sami S. Brandt (2006). “A Generic Camera Model and Calibration Method for Conventional, Wide-Angle, and Fish-Eye Lenses.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 28.8.

Kazhdan, Michael and Hugues Hoppe (2013). “Screened Poisson Surface Reconstruction.” In: *ACM Trans. Graph.* 32.3.

Kettunen, Markus, Marco Manzi, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker (2015). “Gradient-domain Path Tracing.” In: *ACM Trans. Graph.* 34.4.

Kim, Hanme, Stefan Leutenegger, and Andrew J. Davison (2016). “Real-Time 3D Reconstruction and 6-DoF Tracking with an Event Camera.” In: *European Conference on Computer Vision ECCV*.

Knapitsch, Arno, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun (2017). “Tanks and Temples: Benchmarking Large-Scale Scene Reconstruction.” In: *ACM Transactions on Graphics* 36.4.

Kolmogorov, Vladimir and Ramin Zabih (2002). “Multi-camera Scene Reconstruction via Graph Cuts.” In: *Proc. European Conference on Computer Vision 2002 (ECCV 2002)*.

Kopf, Johannes, Fabian Langguth, Daniel Scharstein, Richard Szeliski, and Michael Goesele (2013). “Image-Based Rendering in the Gradient Domain.” In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 32.6.

Laffont, Pierre-Yves, Jong Yun Jun, Christian Wolf, Yu-Wing Tai, Khalid Idrissi, George Drettakis, and Sung-eui Yoon (2010). “Interactive Content-aware Zooming.” In: *Proceedings of Graphics Interface 2010*. GI ’10. Canadian Information Processing Society.

Land, Edwin H. (1977). “The Retinex Theory of Color Vision.” In: *Scientific American* 237.6.

Langguth, Fabian, Kalyan Sunkavalli, Sunil Hadap, and Michael Goesele (2016). “Shading-aware Multi-view Stereo.” In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Lepetit, V., F. Moreno-Noguer, and P. Fua (2009). “EPnP: An Accurate O(n) Solution to the PnP Problem.” In: *International Journal Computer Vision* 81.2.

Levin, A., A. Zomet, and Y. Weiss (2004). “Separating reflections from a single image using local features.” In: *Proc. Computer Vision and Pattern Recognition 2004 (CVPR 2004)* 1.

Levoy, Marc and Pat Hanrahan (1996). “Light field rendering.” In: *Proc. SIGGRAPH ’96*.

Linz, Christian, Christian Lipski, and Marcus Magnor (2010). “Multi-Image Interpolation based on Graph-Cuts and Symmetric Optical Flow.” In: *Proc. Vision, Modeling and Visualization 2010 (VMV 2010)*.

Liu, Feng, Michael Gleicher, Hailin Jin, and Aseem Agarwala (2009). “Content-preserving Warps for 3D Video Stabilization.” In: *ACM SIGGRAPH 2009 Papers*. SIGGRAPH ’09. New Orleans, Louisiana: ACM.

Loop, C. and Z. Zhang (1999). “Computing Rectifying Homographies for Stereo Vision.” In: *Proc. Computer Vision and Pattern Recognition (CVPR ’99)*.

Lowe, David G. (2004). “Distinctive Image Features from Scale-Invariant Keypoints.” In: *International Journal of Computer Vision (IJCV)* 60.2.

MacRobert, T.M. and I.N. Sneddon (1967). *Spherical Harmonics: An Elementary Treatise on Harmonic Functions, with Applications*. International series of monographs in pure and applied mathematics. Pergamon Press.

Mahajan, Dhruv, Fu-Chung Huang, Wojciech Matusik, Ravi Ramamoorthi, and Peter Belhumeur (2009). “Moving gradients: a path-based method for plausible image interpolation.” In: *ACM Transactions on Graphics (Proc. SIGGRAPH 2009)* 28.3.

Marr, D. and T. Poggio (1976). *Cooperative Computation of Stereo Disparity*. Tech. rep. Cambridge, MA, USA.

McMillan, Leonard and Gary Bishop (1995). “Plenoptic Modeling: An Image-based Rendering System.” In: *Proceedings of the 22Nd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’95.

Mostegel, Christian, Rudolf Prettenthaler, Friedrich Fraundorfer, and Horst Bischof (2017). "Scalable Surface Reconstruction from Point Clouds with Extreme Scale and Density Diversity." In: *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Nehab, Diego, Szymon Rusinkiewicz, James Davis, and Ravi Ramamoorthi (2005). "Efficiently Combining Positions and Normals for Precise 3D Geometry." In: *ACM SIGGRAPH 2005 Papers*.

Nicodemus, F. E., J. C. Richmond, J. J. Hsia, I. W. Ginsberg, and T. Limperis (1977). *Geometrical Considerations and Nomenclature for Reflectance*. U.S. Government Printing Office.

Penner, Eric and Li Zhang (2017). "Soft 3D Reconstruction for View Synthesis." In: *ACM Transactions on Graphics (Proc. SIGGRAPH Asia)*. Vol. 36. 6. ACM.

Pérez, Patrick, Michel Gangnet, and Andrew Blake (2003). "Poisson image editing." In: *ACM Transactions on Graphics (Proc. SIGGRAPH 2003)* 22.3.

Popescu, Voicu, Chunhui Mei, Jordan Dauble, and Elisha Sacks (2006). "Reflected-Scene Impostors for Realistic Reflections at Interactive Rates." In: *Computer Graphics Forum (Proc. Eurographics 2006)* 25.3.

Ramamoorthi, Ravi and Pat Hanrahan (2001a). "An Efficient Representation for Irradiance Environment Maps." In: *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '01. New York, NY, USA: ACM.

Ramamoorthi, Ravi and Pat Hanrahan (2001b). "On the relationship between radiance and irradiance: determining the illumination from images of a convex Lambertian object." In: *J. Opt. Soc. Am. A* 18.10.

Ritz, Martin, Fabian Langguth, Manuel Scholz, Michael Goesele, and André Stork (2012). "High Resolution Acquisition of Detailed Surfaces with Lens-Shifted Structured Light." In: *Computers & Graphics* 36.1.

Robertson, Mark A., Sean Borman, and Robert L. Stevenson (2003). "Estimation-theoretic approach to dynamic range enhancement using multiple exposures." In: *Journal of Electronic Imaging* 12.

Salvi, Joaquim, Sergio Fernandez, Tomislav Pribanic, and Xavier Llado (2010). "A State of the Art in Structured Light Patterns for Surface Profilometry." In: *Pattern Recogn.* 43.8.

Scharstein, D. (1994). "Matching images by comparing their gradient fields." In: *Proceedings of the 12th ICPR International Conference on Pattern Recognition*. Vol. 1.

Scharstein, Daniel and Richard Szeliski (2002). "A Taxonomy and Evaluation of Dense Two-Frame Stereo Correspondence Algorithms." In: *Int. J. Comput. Vision* 47.1-3.

Schechner, Y. Y., N. Kiryati, and J. Shamir (2000). “Blind recovery of transparent and semireflected scenes.” In: *Proc. Computer Vision and Pattern Recognition (CVPR 2000)*.

Schechner, Y. Y., J. Shamir, and N. Kiryati (1999). “Polarization-based decorrelation of transparent layers: The inclination angle of an invisible surface.” In: *Proc. International Conference on Computer Vision (ICCV ’99)*.

Schönberger, J. L. and J. M. Frahm (2016). “Structure-from-Motion Revisited.” In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Schönberger, Johannes L., Enliang Zheng, Jan-Michael Frahm, Marc Pollefeys, Jiri Matas, Nicu Sebe, and Max Welling (2016). “Pixelwise View Selection for Unstructured Multi-View Stereo.” In: *ECCV 2016*. Springer International Publishing.

Schöps, Thomas, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger (2017). “A Multi-View Stereo Benchmark with High-Resolution Images and Multi-Camera Videos.” In: *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Seitz, Steve, Brian Curless, James Diebel, Daniel Scharstein, and Richard Szeliski (2006). “A Comparison and Evaluation of Multi-View Stereo Reconstruction Algorithms.” In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’2006)*. IEEE Computer Society.

Seitz, Steven M. and Charles R. Dyer (1999). “Photorealistic Scene Reconstruction by Voxel Coloring.” In: *Int. J. Comput. Vision* 35.2.

Semerjian, Ben (2014). “A New Variational Framework for Multiview Surface Reconstruction.” In: *European Conference on Computer Vision (ECCV)*.

Shade, Jonathan, Steven Gortler, L. He, and Richard Szeliski (1998). “Layered Depth Images.” In: *Proc. SIGGRAPH ’98*.

Shewchuk, Jonathan R (1994). *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Tech. rep. Carnegie Mellon University.

Shi, J. and C. Tomasi (1994). “Good features to track.” In: *IEEE CVPR*.

Shizawa, Masahiko and Keji Mase (1991). “A Unified Computational Theory of Motion Transparency and Motion Boundaries Based on Eigenenergy Analysis.” In: *Proc. Computer Vision and Pattern Recognition (CVPR ’91)*.

Shum, H.-Y.., S.-C. Chan, and S. B. Kang (2007). *Image-Based Rendering*. New York, NY: Springer.

Sinha, Sudipta N., Johannes Kopf, Michael Goesele, Daniel Scharstein, and Richard Szeliski (2012). “Image-based rendering for scenes with reflections.” In: *ACM Transactions on Graphics (Proc. SIGGRAPH 2012)* 31.4.

Sinha, Sudipta N. and Marc Pollefeys (2005). “Multi-View Reconstruction Using Photo-consistency and Exact Silhouette Constraints: A Maximum-Flow Formulation.” In: *Proceedings of the Tenth IEEE International Conference on Computer Vision*. ICCV ’05. IEEE Computer Society.

Sinha, Sudipta N., Drew Steedly, and Richard Szeliski (2009). “Piecewise Planar Stereo for Image-based Rendering.” In: *Proc. International Conference on Computer Vision (ICCV 2009)*.

Snavely, Noah, Steven M. Seitz, and Richard Szeliski (2006). “Photo Tourism: Exploring Photo Collections in 3D.” In: *ACM Transactions on Graphics* 25.3.

Strecha, C., W. von Hansen, L. Van Gool, P. Fua, and U. Thoennessen (2008). “On benchmarking camera calibration and multi-view stereo for high resolution imagery.” In: *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*.

Szeliski, R. (1999). “Prediction error as a quality metric for motion and stereo.” In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2.

Szeliski, Richard (2010). *Computer Vision: Algorithms and Applications*. 1st. Springer-Verlag.

Szeliski, Richard, Shai Avidan, and P. Anandan (2000). “Layer extraction from multiple images containing reflections and transparency.” In: *Proc. Computer Vision and Pattern Recognition (CVPR 2000)*.

Szeliski, Richard, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother (2008). “A Comparative Study of Energy Minimization Methods for Markov Random Fields with Smoothness-Based Priors.” In: *IEEE Trans. Pattern Anal. Mach. Intell.* 30.6.

Triggs, Bill (2004). “Detecting Keypoints with Stable Position, Orientation, and Scale under Illumination Changes.” In: *Computer Vision - ECCV 2004*.

Triggs, Bill, Philip F. McLauchlan, Richard I. Hartley, and Andrew W. Fitzgibbon (2000). “Bundle Adjustment - A Modern Synthesis.” In: *Proceedings of the International Workshop on Vision Algorithms: Theory and Practice*. ICCV ’99. London, UK, UK: Springer-Verlag.

Tsin, Yanghai, Sing Bing Kang, and Richard Szeliski (2006). “Stereo Matching with Linear Superposition of Layers.” In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.2.

Ummenhofer, B. and T. Brox (2015). “Global, Dense Multiscale Reconstruction for a Billion Points.” In: *IEEE International Conference on Computer Vision (ICCV)*.

Vosselman, G. and H.G. Maas (2010). *Airborne and Terrestrial Laser Scanning*. Whittles Publishing.

Waechter, Michael, Mate Beljan, Simon Fuhrmann, Nils Moehrle, Johannes Kopf, and Michael Goesele (2017). “Virtual Rephotography: Novel View Prediction Error for 3D Reconstruction.” In: *ACM Trans. Graph.* 36.1.

Waechter, Michael, Nils Moehrle, and Michael Goesele (2014). “Let There Be Color! Large-Scale Texturing of 3D Reconstructions.” In: *Computer Vision – ECCV 2014*.

Weinmann, Michael, Fabian Langguth, Michael Goesele, and Reinhard Klein (2016). “Advances in Geometry and Reflectance Acquisition.” In: *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics: Tutorials*. Eurographics Association.

Weyrich, Tim, Jason Lawrence, Hendrik P.A. Lensch, Szymon Rusinkiewicz, and Todd Zickler (2009). “Principles of Appearance Acquisition and Representation.” In: *Foundations and Trends in Computer Graphics and Vision* 4.2.

Wilson, Kyle and Noah Snavely (2014). “Robust Global Translations with 1DSfM.” In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Wu, C. (2015). “P3.5P: Pose estimation with unknown focal length.” In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Wu, C., S. Agarwal, B. Curless, and S. M. Seitz (2011a). “Multicore bundle adjustment.” In: *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*.

Wu, Changchang, S. Agarwal, B. Curless, and S. M. Seitz (2011b). “Multicore Bundle Adjustment.” In: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*. CVPR ’11. Washington, DC, USA: IEEE Computer Society.

Wu, Chenglei, B. Wilburn, Y. Matsushita, and C. Theobalt (2011c). “High-quality Shape from Multi-view Stereo and Shading Under General Illumination.” In: *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition*. CVPR ’11.

Wu, Chenglei, Michael Zollhöfer, Matthias Nießner, Marc Stamminger, Shahram Izadi, and Christian Theobalt (2014). “Real-time Shading-based Refinement for Consumer Depth Cameras.” In: *ACM Trans. Graph.* 33.6.

Xu, Di, Qi Duan, Jianming Zheng, Juyong Zhang, Jianfei Cai, and Tat-Jen Cham (2014). “Recovering Surface Details under General Unknown Illumination Using Shading and

Coarse Multi-view Stereo." In: *IEEE Conference on Computer Vision and Pattern Recognition*.

Yoon, Kuk-Jin and In So Kweon (2006). "Adaptive Support-Weight Approach for Correspondence Search." In: *IEEE Trans. Pattern Anal. Mach. Intell.* 28.4.

Yu, L. F., S. K. Yeung, Y. W. Tai, and S. Lin (2013). "Shading-Based Shape Refinement of RGB-D Images." In: *Computer Vision and Pattern Recognition (CVPR)*.

Zhang, Guo-Xin, Ming-Ming Cheng, Shi-Min Hu, and Ralph R. Martin (2009). "A Shape-Preserving Approach to Image Resizing." In: *Computer Graphics Forum*.

Zhou, Z., Z. Wu, and P. Tan (2013). "Multi-view Photometric Stereo with Spatially Varying Isotropic Materials." In: *Computer Vision and Pattern Recognition (CVPR)*.

Zitnick, C. Lawrence, Sing Bing Kang, Matthew Uyttendaele, Simon Winder, and Richard Szeliski (2004). "High-quality video view interpolation using a layered representation." In: *ACM Transactions on Graphics (Proc. SIGGRAPH 2004)* 23.3.

Zollhöfer, Michael, Angela Dai, Matthias Innmann, Chenglei Wu, Marc Stamminger, Christian Theobalt, and Matthias Nießner (2015). "Shading-based Refinement on Volumetric Signed Distance Functions." In: *ACM Transactions on Graphics (TOG)*.

Publications (co-)authored by Fabian Langguth

Ackermann, Jens, **Fabian Langguth**, Simon Fuhrmann, and Michael Goesele (2012). "Photometric Stereo for Outdoor Webcams." In: *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ackermann, Jens, **Fabian Langguth**, Simon Fuhrmann, Arjan Kuijper, and Michael Goesele (2014). "Multi-view Photometric Stereo by Example." In: *3D Vision (3DV), 2014 2nd International Conference on* 1.

Aroudj, Samir, Patrick Seemann, **Fabian Langguth**, Stefan Guthe, and Michael Goesele (2017). "Visibility-Consistent Thin Surface Reconstruction Using Multi-Scale Kernels." In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)*. Vol. 36. 6. ACM.

Fuhrmann, Simon, **Fabian Langguth**, and Michael Goesele (2014). "MVE - A Multiview Reconstruction Environment." In: *Proceedings of the Eurographics Workshop on Graphics and Cultural Heritage (GCH)*. Vol. 6. 7.

Fuhrmann, Simon, **Fabian Langguth**, Nils Moehrle, Michael Waechter, and Michael Goesele (2015). "MVE - An image-based reconstruction environment." In: *Computers & Graphics* 53.

Goesele, Michael, Jens Ackermann, Simon Fuhrmann, Ronny Klowsky, **Fabian Langguth**, Patrick Muecke, and Martin Ritz (2010b). "Scene Reconstruction from Community Photo Collections." In: *IEEE Computer* 43.6.

Kopf, Johannes, **Fabian Langguth**, Daniel Scharstein, Richard Szeliski, and Michael Goesele (2013). "Image-Based Rendering in the Gradient Domain." In: *ACM Transactions on Graphics (Proceedings of SIGGRAPH Asia)* 32.6.

Langguth, **Fabian** and Michael Goesele (2013). "Guided Capturing of Multi-view Stereo Datasets." In: *Proceedings of Eurographics 2013 Short Papers*.

Langguth, **Fabian**, Kalyan Sunkavalli, Sunil Hadap, and Michael Goesele (2016). "Shading-aware Multi-view Stereo." In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Ritz, Martin, **Fabian Langguth**, Manuel Scholz, Michael Goesele, and André Stork (2012). "High Resolution Acquisition of Detailed Surfaces with Lens-Shifted Structured Light." In: *Computers & Graphics* 36.1.

Weinmann, Michael, **Fabian Langguth**, Michael Goesele, and Reinhard Klein (2016). "Advances in Geometry and Reflectance Acquisition." In: *Proceedings of the 37th Annual Conference of the European Association for Computer Graphics: Tutorials*. Eurographics Association.

Supplementary Video

This thesis has a supplementary video with results from the rendering methods presented in Chapter 4 and Chapter 5. In the printed version the video is on the attached USB flash drive. For the digital version the video is available via a download link on the publication website.